Comprehensive Network Traffic Analysis on Mikrotik RouterOS v7: A Guide to Identifying High-Bandwidth Users and Applications

1. Introduction: Illuminating Your Network Traffic

Understanding network traffic patterns is essential for maintaining an efficient, secure, and well-performing home network, particularly for technically adept users. Monitoring allows for the identification of devices or applications consuming disproportionate amounts of bandwidth, aids in troubleshooting performance bottlenecks, and enhances awareness of overall network activity, which can have security implications. This guide directly addresses the need to identify high-bandwidth IP addresses on a local network using a Mikrotik RouterOS v7 system and, more importantly, to understand the underlying reasons—such as specific protocols and applications—for this data consumption.

The focus of this document is to provide a range of solutions, from quick on-router checks to more comprehensive off-router analysis leveraging flow data. Priority is given to free and open-source methods, acknowledging the user's technical proficiency and willingness to deploy software on auxiliary servers or containers. Both passive and active monitoring techniques will be explored.

A critical aspect of the user's request is the desire for protocol and application breakdowns. Simply identifying a high-traffic IP address often falls short of providing actionable information. Knowing *what* services or applications are responsible for the traffic allows for more informed decisions, whether for optimizing Quality of Service (QoS), addressing misconfigured or overly verbose applications, or simply satisfying a technical curiosity about network behavior. This emphasis on the "why" behind traffic consumption underscores the importance of tools capable of deeper inspection or sophisticated flow analysis, which will be a recurring theme.

2. RouterOS v7: Built-in Monitoring Capabilities

Mikrotik's RouterOS provides several tools for network monitoring directly on the router. However, with the release of version 7, there have been significant changes to how traffic accounting is handled, alongside the continued availability of real-time inspection tools.

2.1 The Paradigm Shift: IP Accounting's Departure and Traffic Flow's Ascendance

A notable change in RouterOS v7 is the removal of the traditional IP Accounting

feature.[1] This feature, familiar to users of older RouterOS versions, provided a simple way to track byte and packet counts per IP address directly on the router. Its removal signifies a shift in how Mikrotik approaches traffic statistics.[1]

The primary mechanism now recommended for detailed traffic statistics export and subsequent analysis is /ip traffic-flow. This feature allows the router to export NetFlow (versions 1, 5, 9) or IPFIX (NetFlow version 10) data to an external collector.[3] While this necessitates an external system for data collection and analysis, it aligns RouterOS with industry-standard protocols. This "forced modernization" towards NetFlow/IPFIX, though potentially requiring a new workflow for some, ultimately offers enhanced flexibility and power. These standardized protocols are supported by a wide array of sophisticated third-party analysis tools, many of which are open-source and well-suited to the detailed analysis desired, including protocol and application breakdowns.

2.2 Quick Checks: Leveraging Torch for Real-time Interface Monitoring

For immediate, real-time analysis of traffic passing through an interface, RouterOS offers the Torch tool (/tool torch).[5] Torch provides a dynamic, live view of traffic flows, making it useful for on-the-spot troubleshooting and identifying active high-bandwidth consumers.

Basic Usage:
To start Torch, one typically specifies the interface to monitor. For instance, to monitor all traffic passing through the main LAN bridge (often named bridge1 or bridge-lan):

Code snippet

```
/tool torch interface=bridge1
```

Or for a specific Ethernet port:

Code snippet

```
/tool torch interface=ether1
```

Filtering Capabilities:

Torch allows for filtering to narrow down the observed traffic by various parameters, including source or destination IP address, protocol (e.g., TCP, UDP, ICMP), and specific ports.5 This is crucial for focusing on particular devices or services.
Example: To monitor HTTPS traffic (TCP port 443) from a specific internal IP address (e.g., 192.168.1.10) on bridge1:

Code snippet

```
/tool torch interface=bridge1 src-address=192.168.1.10/32 protocol=tcp port=443
```

Interpreting Output:
The Torch output displays several columns, including Src. Address, Dst. Address, Protocol, Src. Port, Dst. Port, TX Rate (transmit rate), and RX Rate (receive rate).5 By observing the TX Rate and RX Rate columns, one can quickly identify which connections or IPs are currently transferring the most data.
Limitations:
It is important to understand Torch's limitations. It provides a snapshot of live traffic and is not designed for historical data logging or long-term analysis.5 Furthermore, Torch can be resource-intensive, especially when run without specific filters on high-traffic interfaces, potentially impacting router performance.5 While Torch is an excellent first-response tool for answering "What is happening on my network right now?", it does not fulfill the need for persistent data collection, historical trending, or in-depth application-level analysis required for a comprehensive understanding of bandwidth usage.

2.3 Detailed Captures: Using Packet Sniffer and Mangle for In-Depth Inspection

For more granular analysis, RouterOS includes a packet sniffer tool (/tool sniffer) that can capture full packets to a file or stream them to a remote host for analysis with tools like Wireshark.[7]

Capturing to a File:
Packets can be captured directly on the router and saved to a file.

- **Commands:** To start a capture on ether1 and save to mycapture.pcap:
  Code snippet
  ```
  /tool sniffer start file-name=mycapture.pcap interface=ether1
  ```
  To stop the capture:
  Code snippet
  ```
  /tool sniffer stop
  ```
  If a file-name was not specified at the start, captured packets (held in memory for a limited time) can be saved using [7]:
  Code snippet
  ```
  /tool sniffer save file-name=mycapture.pcap
  ```

- **Limits:** file-limit (maximum file size) and memory-limit (buffer size in RAM) can be set to manage resource usage.[7]
- **Filtering:** The sniffer supports various filters, including IP address, port, protocol, MAC address, and traffic direction (rx or tx) to narrow down the scope of captured packets.[7]
- **Downloading:** Captured files can be downloaded from the router using WinBox (Files menu), SFTP, or FTP.[7] These files can then be opened in Wireshark on a macOS desktop or other systems for detailed protocol-level analysis.

Streaming to Wireshark via TZSP:
Instead of saving to a file on the router, packets can be streamed live to a remote machine running Wireshark using the Tazmen Sniffer Protocol (TZSP).

- **Using /tool sniffer:** Enable streaming and specify the target IP (e.g., the macOS desktop's IP) and port (default TZSP port is 37008) [7]:
  Code snippet
  ```
  /tool sniffer set streaming-enabled=yes streaming-server=<macOS_IP_address>
  /tool sniffer start
  ```
  Wireshark on the macOS machine needs to be configured to listen for incoming TZSP streams on UDP port 37008.[10]
- **Using Mangle sniff-tzsp Action:** For more targeted streaming, the firewall mangle facility offers a sniff-tzsp action. This allows specific traffic, matched by firewall rules, to be sent to a TZSP receiver.[10] This is more efficient than sniffing an entire interface if only particular flows are of interest. Example: To stream packets from a specific local IP (e.g., 192.168.1.50) to a Wireshark instance on 192.168.1.100:
  Code snippet
  ```
  /ip firewall mangle
  add chain=prerouting action=sniff-tzsp sniff-target=192.168.1.100
  sniff-target-port=37008 src-address=192.168.1.50
  add chain=postrouting action=sniff-tzsp sniff-target=192.168.1.100
  sniff-target-port=37008 dst-address=192.168.1.50
  ```
  (Note: Capturing both directions relative to the host of interest often requires two rules if using src/dst address as primary matcher).

Resource Considerations:
Full packet capture is inherently resource-intensive. Capturing all traffic on a busy interface can significantly load the router's CPU and fill storage or memory buffers quickly.7 Using filters is highly recommended.
Packet sniffing provides the most detailed view of network traffic, offering ground

truth at the packet level. However, analyzing large capture files or continuous streams to identify aggregated top talkers or application usage trends over time is cumbersome and inefficient. While invaluable for forensic analysis or debugging specific communication issues once a problematic flow is identified, it's not the primary tool for the continuous, aggregated monitoring the user seeks. The sniff-tzsp mangle action offers a good balance for selectively sending specific, interesting flows to Wireshark without the overhead of capturing everything.

2.4 Table: Mikrotik RouterOS Monitoring Tools Overview

To summarize the on-router tools:

Tool	Primary Use Case	Data Granularity	Router Resource Impact	Suitability for User's Goals
Torch	Real-time interface stats	Flow-level summary	High for broad use	Quick checks for live activity only
Packet Sniffer	Deep packet capture/streaming	Full packet	Med-High (depends on filters/rate)	Forensic/deep-dive; not for aggregated reporting
Traffic Flow	Bulk flow data export for analysis	Flow metadata	Low-Med for export	**Core data source for offloaded comprehensive analysis**
Mangle L7	L7 pattern matching in firewall	Connection-level	Very High	Limited/problematic for reporting; primarily a firewall tool

This overview highlights that for comprehensive, historical analysis with protocol and application breakdowns, /ip traffic-flow is the most suitable on-router mechanism, as it provides the necessary data for offloaded processing.

3. Comprehensive Traffic Analysis with NetFlow/IPFIX

For a detailed and historical understanding of network traffic, especially to identify top consumers and the nature of their traffic, exporting flow data using NetFlow or IPFIX is the recommended approach with RouterOS v7.

3.1 Understanding NetFlow/IPFIX: The Foundation for Rich Analysis

NetFlow (developed by Cisco) and IPFIX (IP Flow Information Export, an IETF standard based on NetFlow v9) are protocols designed to collect and export information about IP traffic flows passing through a network device.[3] A "flow" is generally defined as a unidirectional sequence of packets sharing common characteristics, such as:

- Source and Destination IP addresses
- Source and Destination Ports (for TCP/UDP)
- IP Protocol type (e.g., TCP, UDP, ICMP)
- Type of Service (ToS) byte (which includes DSCP values)
- Input/Output logical interface

When a router is configured for NetFlow/IPFIX export, it observes packets, groups them into flows, and periodically exports records containing metadata about these flows to a designated collector. This metadata typically includes the identifying characteristics listed above, along with byte and packet counts for the flow, start and end timestamps, and potentially other information elements (IEs). IPFIX is particularly extensible, allowing vendors to include custom IEs.[4]

For the user's goal of identifying high-bandwidth IPs and understanding *why* they are high-bandwidth, flow data is superior to simple packet counting or even real-time tools like Torch. It provides a structured dataset that details *who* is communicating with *whom*, *what* protocols are being used, and *how much* data is being exchanged, all without the overhead of capturing and storing every single packet. This aggregated, yet detailed, information is ideal for feeding into analysis and visualization tools.

3.2 Configuring Traffic Flow (NetFlow v9/IPFIX) on Your Mikrotik Router

RouterOS v7 supports NetFlow versions 1, 5, 9, and IPFIX (referred to as version 10 in some contexts).[3] For modern collectors, IPFIX or NetFlow v9 are preferred due to their flexibility and richer data sets. ElastiFlow documentation, for example, shows configuration with version=10 (IPFIX) [4], while other general NetFlow guides might show version=9.[3] The choice depends on the capabilities of the selected collector software.

Step-by-Step CLI Configuration:

1. Enable Traffic Flow:
   This globally enables the traffic flow feature on the router.
   Code snippet
   `/ip traffic-flow set enabled=yes`

   ([3])
2. Specify Interfaces to Monitor:
   You can monitor all interfaces or select specific ones. For typical home network monitoring, tracking traffic on the LAN bridge interface (e.g., bridge-lan or bridge1) is usually most relevant as it captures all traffic from/to local devices.
   To monitor all interfaces:
   Code snippet
   `/ip traffic-flow set interfaces=all`

   To monitor a specific bridge interface named bridge-lan:
   Code snippet
   `/ip traffic-flow set interfaces=bridge-lan`

   ([3])
3. Configure Target Collector:
   This tells the router where to send the flow records. Replace <collector_IP> with the IP address of the server or container running your flow collector software, and <collector_port> with the UDP port the collector is listening on (common ports include 2055, 4739, 9995).
   For IPFIX (recommended if collector supports it):
   Code snippet
   `/ip traffic-flow target add dst-address=<collector_IP> port=4739 version=10`

   For NetFlow v9:
   Code snippet
   `/ip traffic-flow target add dst-address=<collector_IP> port=2055 version=9`

   ([3])

Key Parameters and Their Importance:
The behavior of flow export is influenced by several timeout and cache settings. These need to be configured appropriately to ensure accurate and timely data.

- active-flow-timeout: Defines how frequently the router exports an update for flows that are still active (long-lived connections). A common value is 1m (1 minute) or 5m. Shorter timeouts provide more granular updates but increase the volume of flow data sent to the collector.[3] For home use, 1m is often a good

balance.
- inactive-flow-timeout: Specifies how long the router waits after the last packet of a flow is seen before exporting the flow record and removing it from the active flow cache. Common values are 15s (15 seconds) or 30s. Shorter timeouts mean quicker reporting of completed flows.[3] 15s is generally suitable.
- cache-entries: Sets the maximum number of concurrent flows the router can track in its cache. If this limit is exceeded, new flows might not be tracked, or older ones might be prematurely expired, leading to inaccurate data. Values like 4k (4096) or 8k (8192) are common starting points.[3] For a home network, 4k or 8k should generally be sufficient unless there's an unusually high number of concurrent connections (e.g., heavy P2P use).

Example setting these parameters:

Code snippet

```
/ip traffic-flow set active-flow-timeout=1m inactive-flow-timeout=15s
cache-entries=8k
```

The quality and completeness of the data analyzed by the collector are entirely dependent on these RouterOS configurations. If timeouts are too long, the view of traffic will be delayed. If the cache is too small for the network's activity, flows will be missed.

Verification Commands:
After configuration, verify the settings:

Code snippet

```
/ip traffic-flow print
/ip traffic-flow target print
```

([3]) These commands will display the current Traffic Flow settings and the configured targets, allowing confirmation that the router is set up to export data as intended.

4. Free & Open-Source Flow Collection and Analysis Solutions

Once your Mikrotik router is configured to export NetFlow/IPFIX data, you need a system to collect, store, and analyze these flows. The following solutions are free, open-source, and can be deployed on a separate server, a virtual machine, or within containers. The user's macOS desktop can also run some of these, particularly if containerized, though an always-on system is preferable for continuous monitoring.

4.1 Solution A: ntopng with netflow2ng (Cost-Effective & Feature-Rich for Home Use)

This combination offers a powerful suite for traffic analysis, making ntopng's advanced features accessible for home users leveraging Mikrotik's NetFlow v9 export.

Introduction to ntopng:
ntopng is a popular open-source, web-based network traffic monitoring application.14 Its key features relevant to this guide include:

- Real-time and historical traffic analysis.
- Identification of top talkers (by IP address, Autonomous System, etc.).
- Application protocol identification using its integrated nDPI (NetFlow Deep Packet Inspection) library, which can often see beyond simple port numbers.[14]
- Geographical mapping of traffic.
- Alerting capabilities. The Community Edition of ntopng is free and open source.[14]

The nProbe Dilemma and netflow2ng as a Solution:
Traditionally, ntopng has relied on nProbe for collecting and preprocessing NetFlow/IPFIX data before forwarding it to ntopng via a ZMQ (ZeroMQ) message queue.18 While nProbe is highly capable, its standard licenses are commercial (though free licenses are available for non-profit and academic institutions).20 This cost can be a barrier for home users.
netflow2ng is a free, open-source tool specifically created as an alternative for collecting NetFlow v9 data (which Mikrotik RouterOS readily exports) and publishing it in a format that ntopng Community Edition can consume via ZMQ.[22] It is explicitly targeted at home and SOHO (Small Office/Home Office) users [22], effectively bridging the gap for those who want to use ntopng with NetFlow v9 sources without purchasing nProbe. This makes the powerful analysis capabilities of ntopng, especially its nDPI engine for application identification, accessible for this use case.

Setup Guide:

1. **Installing ntopng Community Edition:**
    - It's recommended to install ntopng on a Linux server or in a Docker container. The user's server or container host would be suitable.

- For Ubuntu/Debian, instructions typically involve adding the official ntop repository and then installing the ntopng package along with its dependencies, such as redis (used by ntopng for caching and short-term data storage).[23]

  ```
  Bash
  # Example steps for Ubuntu (refer to official ntop.org documentation for current commands)
  wget https://packages.ntop.org/apt/22.04/all/apt-ntop.deb # Adjust for your OS version
  sudo apt install./apt-ntop.deb
  sudo apt update
  sudo apt install ntopng redis-server
  ```

- Basic ntopng configuration is done in /etc/ntopng/ntopng.conf. Key parameters include the network interface(s) ntopng should listen on (though for netflow2ng, this will be a ZMQ endpoint) and the web server port (default is 3000).

2. **Installing and Configuring netflow2ng:**
   - netflow2ng can be built from its Go source code or run from a Docker image.[22] The Docker approach is generally simpler.

     ```
     Bash
     # Example: Clone and build from source
     git clone https://github.com/synfinatic/netflow2ng.git
     cd netflow2ng
     make
     # The binary will be in the dist/ directory
     ```

     Or, using Docker:

     ```
     Bash
     # Refer to netflow2ng or community Docker Hub images
     # Example: docker run -d --name netflow2ng -p 2055:2055/udp -p 5556:5556/tcp some-netflow2ng-image <options>
     ```

   - **Key netflow2ng configuration options** (typically passed as command-line arguments, run netflow2ng -h for a full list [22]):
     - NetFlow listening address and port: e.g., -listen-netflow :2055 (to match the Mikrotik export configuration).
     - ZMQ publisher endpoint: e.g., -publish-zmq tcp://127.0.0.1:5556 (if ntopng is on the same host) or -publish-zmq tcp://<IP_of_ntopng_host>:5556.
     - The GitHub repository ThePlexus/ntopng-docker [25] provides a Docker setup that bundles ntopng and netflow2ng, which could simplify deployment. It specifies UDP port 2055 for NetFlow collection.
3. **Connecting Mikrotik NetFlow to netflow2ng:**

- On the Mikrotik router, ensure the /ip traffic-flow target configuration has:
  - dst-address: The IP address of the host running netflow2ng.
  - port: The UDP port netflow2ng is listening on (e.g., 2055).
  - version=9.
- Example Mikrotik configuration snippet from ntop.org documentation for sending to nProbe (adapt IP and port for netflow2ng) [18]:
  Code snippet
  ```
  /ip traffic-flow
  set enabled=yes active-flow-timeout=1m inactive-flow-timeout=15s
  /ip traffic-flow target
  add dst-address=<netflow2ng_server_IP> port=2055 version=9
  v9-template-timeout=1m
  ```

4. **Connecting netflow2ng to ntopng:**
   - Modify the ntopng startup command or its configuration file (/etc/ntopng/ntopng.conf or /etc/ntopng/ntopng.d/ files) to specify the ZMQ interface provided by netflow2ng.
   - The interface string should be -i tcp://<netflow2ng_zmq_ip>:<netflow2ng_zmq_port>.
   - Example from ntop.org (originally for nProbe, adaptable for netflow2ng) [18]: If netflow2ng publishes on tcp://127.0.0.1:5556:
     Bash
     ```
     # In ntopng.conf or as a command line argument to ntopng:
     # -i=tcp://127.0.0.1:5556
     ```
     Then restart the ntopng service.

Visualizing Data in ntopng:

- Access the ntopng web interface, typically at http://<ntopng_server_IP>:3000.
- The main dashboard provides an overview. Look for sections or tabs related to:
  - **Flows:** Allows browsing and filtering of individual network flows.
  - **Hosts:** Shows statistics per local and remote host, including traffic volumes.
  - **Top Talkers:** Often presented as Sankey diagrams or pie charts showing the hosts and connections generating the most traffic.[16]
  - **Applications:** Displays traffic breakdown by application protocol, identified by nDPI.[16]
- ntopng's interface is interactive, allowing users to click on hosts, flows, or applications to drill down for more detailed information.

Limitations:
The primary limitation of netflow2ng is its focus on NetFlow v9.22 If advanced IPFIX-specific

Information Elements are crucial, this might be a constraint. However, for identifying top talkers by IP and getting application/protocol breakdowns, NetFlow v9 often provides sufficient data.

4.2 Solution B: ElastiFlow with Elasticsearch/OpenSearch (Powerful & Scalable, Visualizations)

ElastiFlow offers a robust solution for collecting and visualizing flow data, leveraging the power of Elasticsearch or OpenSearch for data storage and Kibana or OpenSearch Dashboards for visualization. This combination is known for its scalability and rich analytical capabilities.

Introduction to ElastiFlow Unified Flow Collector:
The ElastiFlow Unified Flow Collector is designed to receive, decode, transform, normalize, translate, and enrich network flow records (including NetFlow, IPFIX, and sFlow) and other telemetry.27 It can output the processed data to various platforms, with Elasticsearch and OpenSearch being common choices for use with ElastiFlow's pre-built dashboards.27
Licensing: Community and Basic (Free) Tiers:
ElastiFlow offers free tiers suitable for home or personal use:

- **Community Tier:** This tier operates without a license key. It is limited to processing 500 flow records per second (FPS).[30] The number of supported Information Elements (IEs) from flow records is also limited (e.g., 83 for IPFIX, 152 for NetFlow in one version of documentation [30]).
- **Basic Tier:** This tier is also free but requires registration on the ElastiFlow website to obtain a license key.[30] It shares the same 500 FPS limit as the Community tier.[30] The Basic tier supports a significantly larger set of standard Information Elements compared to the Community tier (e.g., 450 for IPFIX, 519 for NetFlow [30]).

For most home networks, the 500 FPS limit of the Community or Basic tier should be more than sufficient. The increased IE support in the Basic tier makes it a preferable free option. These free tiers provide a substantial amount of functionality for personal use.[33]

Setup Guide (Docker Recommended):
ElastiFlow provides comprehensive documentation for deploying its stack, often using Docker and Docker Compose, which simplifies the setup of the collector, Elasticsearch/OpenSearch, and Kibana/OpenSearch Dashboards.34

1. Deploying Elasticsearch & Kibana (or OpenSearch & OpenSearch Dashboards): These can be run as Docker containers. ElastiFlow's documentation often includes docker-compose.yml files that set up the entire backend stack.34 Ensure sufficient resources (RAM, CPU, disk space) are allocated, especially for Elasticsearch/OpenSearch. A system with 16GB RAM is suggested for a 500 FPS load.34

2. **Deploying ElastiFlow Unified Flow Collector:**
    - The official Docker image is elastiflow/flow-collector.[36]
    - Configuration is typically managed via environment variables passed to the Docker container or through YAML configuration files mounted into the container.
    - Key environment variables for basic setup include:
        - EF_FLOW_INPUT_UDP_LISTEN_ADDRESS: IP address to listen on (e.g., 0.0.0.0 for all interfaces).
        - EF_FLOW_INPUT_UDP_LISTEN_PORT: UDP port for incoming flows (e.g., 4739 for IPFIX, or 2055, 9995). This must match the Mikrotik target port.
        - EF_OUTPUT_ELASTICSEARCH_ENABLE=true (or EF_OUTPUT_OPENSEARCH_ENABLE=true).
        - EF_OUTPUT_ELASTICSEARCH_HOSTS: Comma-separated list of Elasticsearch hosts (e.g., http://elasticsearch_host:9200).
        - If using the Basic tier license, EF_ACCOUNT_ID and EF_FLOW_LICENSE_KEY must be set.[37]
        - EF_LICENSE_ACCEPTED=true to accept the EULA.[37]
    - A docker-compose.yml example from ElastiFlow's GitHub or documentation would typically include services for the collector, Elasticsearch, and Kibana.[34]
3. **Configuring Mikrotik IPFIX for ElastiFlow:**
    - On the Mikrotik router, configure /ip traffic-flow target to use version=10 (for IPFIX).
    - Set dst-address to the IP of the host running the ElastiFlow collector.
    - Set port to the UDP port ElastiFlow is configured to listen on (e.g., 4739).[4]
    - Adjust active-flow-timeout (e.g., 1m) and inactive-flow-timeout (e.g., 15s) as needed.[4]
4. Data Enrichment (Optional but Recommended):
   ElastiFlow can enrich flow data with GeoIP (geographic location of public IPs) and ASN (Autonomous System Number) information. This requires MaxMind GeoLite2 database files, which can be obtained for free with registration. These database files need to be made available to the ElastiFlow collector container, typically via a Docker volume mount.34 Environment variables like EF_PROCESSOR_ENRICH_IPADDR_MAXMIND_GEOIP_ENABLE=true control these features.

Visualizing Data with ElastiFlow Dashboards:

- Once the stack is running and receiving flow data, import the pre-built ElastiFlow dashboards into Kibana or OpenSearch Dashboards. These are usually provided as .ndjson files.[34]

- ElastiFlow dashboards are comprehensive and provide views for:
  - Top Talkers (by IP, conversations, applications)
  - Application Flows (based on port, and potentially enriched application names)
  - Sankey diagrams illustrating traffic relationships
  - GeoIP maps showing traffic origins and destinations
  - Security-related views (e.g., traffic to/from known malicious IPs if threat intelligence is integrated) ([38])

The combination of a dedicated, efficient flow collector and professionally designed dashboards makes ElastiFlow a very appealing option for users who want powerful analytics and visualizations without the need to build them from scratch. The free tiers are quite capable for home network monitoring. Application visibility is a strong point, leveraging both standard flow fields and potential enrichments.[38]

4.3 Solution C: Prometheus & Grafana (Metrics-Focused, More DIY for Detailed Flow Analysis)

Prometheus and Grafana form a popular open-source stack for metrics monitoring and visualization. While extremely powerful for time-series data and alerting, using them for detailed NetFlow/IPFIX analysis (like identifying specific high-bandwidth flows or deep application breakdowns) requires a more "do-it-yourself" approach compared to ntopng or ElastiFlow.

Overview:

- **Prometheus:** A time-series database and monitoring system. It scrapes metrics from configured endpoints.[45]
- **Grafana:** A visualization platform that can use Prometheus (and many other sources) as a data source to create rich dashboards.[45]

Options for Exporting NetFlow/IPFIX Data to Prometheus:
Directly ingesting raw NetFlow/IPFIX records into Prometheus is not its typical use case. Instead, an intermediary "exporter" or collector is needed to process the flow data and expose relevant metrics that Prometheus can scrape.

1. **netflow2ng Prometheus Metrics:**
   - The netflow2ng tool, discussed earlier for its ZMQ output to ntopng, also explicitly lists "Prometheus metrics" as a feature.[22]
   - To use this, one would need to identify the command-line flag or configuration option in netflow2ng that enables the Prometheus metrics endpoint, and the port/path it exposes (e.g., often /metrics on a specific port like 9100 or a custom one). This information should be available by running netflow2ng -h.[22] General Prometheus exporter documentation [48] describes

common patterns but not netflow2ng specifics.
    - Prometheus would then be configured to scrape this endpoint.
2. **pmacct with a Prometheus Exporter:**
    - pmacct is a suite of network monitoring tools. Its nfacctd daemon can collect NetFlow/IPFIX data.[54]
    - pmacct can aggregate this data and store it in various backends (databases, Kafka, flat files) or publish it via AMQP/Kafka.[54]
    - While pmacct itself may not have a built-in, direct Prometheus exporter for all its aggregated flow data, the data it collects can be fed into a system that does. For example:
        - If pmacct writes to a database (e.g., PostgreSQL, MySQL), a separate Prometheus exporter for that database type could be used to query aggregated statistics and expose them as metrics.
        - If pmacct sends data to Kafka, a tool like neptune-networks/flow-exporter [45] could consume from Kafka and expose Prometheus metrics. This adds Kafka to the pipeline, increasing complexity.
    - The pmacct project focuses on data collection and flexible output, enabling integration with various analysis platforms.[47]

Setup Considerations and Complexity:

- This approach is generally more focused on aggregated metrics (e.g., total bytes per IP, traffic per protocol over time) rather than interactive exploration of individual flow records.
- The user would need to define which metrics are important to extract from the flow data and configure the exporter and Prometheus/Grafana accordingly.
- Application protocol identification would depend on the capabilities of the chosen NetFlow collector/exporter (e.g., if pmacct is used with nDPI for classification, those classified flows could be aggregated). netflow2ng relies on goflow, whose application identification capabilities would need to be checked.

Basic Grafana Dashboards:
Once Prometheus is scraping metrics from the chosen exporter, Grafana dashboards can be built to visualize:

- Traffic rates (bytes/packets per second) for top source/destination IPs.
- Protocol distribution over time.
- Network interface utilization based on flow data. Grafana offers flexibility in dashboard creation but lacks the out-of-the-box, flow-specific browsing capabilities of ntopng or the pre-built dashboards of ElastiFlow.

This stack is excellent for understanding trends in "how much" and "when" for

predefined, aggregated metrics. However, for the deep-dive "why" into specific connections or for rich, interactive application-level breakdowns without significant custom query and dashboard development, ntopng or ElastiFlow are generally more direct solutions. If netflow2ng is already being used for ntopng, its Prometheus endpoint could be a convenient way to get some basic flow-derived metrics into an existing Prometheus/Grafana setup as a complementary view.

5. Understanding Application and Protocol Breakdowns

A key requirement is to understand not just *which* IPs transfer the most data, but *why*—specifically, which protocols and applications are responsible. This level of detail often requires more than simple port-based identification.

5.1 The Power of Deep Packet Inspection (DPI) and Flow Enrichment

Simply looking at TCP/UDP port numbers to identify applications is increasingly unreliable. Many applications use common ports (e.g., HTTPS/443 for a vast array of web services) or non-standard ports to evade simple firewalls. True application visibility often relies on Deep Packet Inspection (DPI) techniques or sophisticated flow data enrichment.

- nDPI in ntopng:
  ntopng, including its free Community Edition, integrates nDPI (ntop Deep Packet Inspection), an open-source library capable of identifying application protocols by analyzing traffic patterns, signatures, and metadata within flows.14 This goes far beyond port numbers. nDPI can:
  - Detect known protocols on non-standard ports.
  - Identify applications that might be tunneled over standard ports (e.g., various services over HTTPS).
  - Provide some visibility into encrypted traffic, for instance, by analyzing SSL/TLS certificate information or other unencrypted metadata exchanged during session setup.[58] This capability is a significant advantage for understanding the "application" part of the user's query.
- ElastiFlow's Application Identification:
  ElastiFlow also provides mechanisms for application visibility.38 This can come from several sources:
  - **IPFIX Option Templates:** If the Mikrotik router exports application identifiers via IPFIX Option Templates (e.g., applicationTag or similar vendor-specific elements), ElastiFlow can decode and display this information.[28] The availability of such elements depends on RouterOS's IPFIX implementation.
  - **Enrichment:** ElastiFlow can enrich flow records with application names based

on its own internal mappings or integrations. The ElastiFlow NetIntel service, for example, can provide application/service names and threat intelligence data.[27] Some advanced enrichment features or IE support might be tied to specific license tiers.[30]
    - The combination of direct flow data and external enrichment allows ElastiFlow to provide a comprehensive view of application traffic.[39]

5.2 Mikrotik RouterOS Layer 7 Inspection: Capabilities and Severe Limitations for Reporting

RouterOS includes a Layer 7 (L7) protocol matcher within its firewall functionality.[60] This feature uses POSIX regular expressions (regex) to search for patterns within the payload of TCP, UDP, or ICMP streams.

How it Works:
The L7 matcher collects the first 10 packets of a connection or the first 2 kilobytes (KB) of data from a connection, whichever comes first. It then attempts to match the configured regex patterns against this collected data.60 If a pattern is found, a corresponding firewall rule can take action (e.g., accept, drop, mark).

Severe Limitations for Comprehensive Traffic Analysis and Reporting:

- **Extremely Resource Intensive:** The L7 matcher is notoriously demanding on CPU and memory resources.[60] Using it for generic traffic inspection or with many L7 rules is strongly discouraged as it can easily overwhelm the router and degrade network performance. It is intended for very specific, targeted traffic.
- **Limited Inspection Depth:** Because it only inspects the initial part of a connection (10 packets or 2KB), if the unique pattern identifying an application appears later in the data stream, the L7 matcher will miss it and classify the protocol as unknown.[60]
- **Ineffective for Encrypted Traffic:** The L7 matcher cannot inspect encrypted payloads.[60] Since a vast majority of internet traffic is now encrypted (HTTPS, VPNs, secure messaging apps, etc.), the L7 matcher is blind to the actual application content within these streams. It might be able to identify the SSL/TLS handshake itself using a regex, but not the application data being tunneled. For example, it cannot distinguish between different websites or services accessed over HTTPS.
- **Requires Seeing Both Traffic Directions:** For accurate pattern matching, the L7 matcher generally needs to see both incoming and outgoing packets of a connection. This often means placing L7 rules in the forward chain, or if in input/prerouting, also in output/postrouting chains, adding to configuration complexity.[60]

- **Regex Complexity and Maintenance:** Crafting effective and efficient regex patterns for diverse applications is challenging and requires ongoing maintenance as application protocols evolve. Example L7 patterns can be found on the l7-filter project page, but their effectiveness can vary.[60]

Use Cases:
The RouterOS L7 matcher is primarily a firewall tool for very specific, limited use cases, such as attempting to identify or block simple, unencrypted protocols based on unique header patterns. It is not suitable for comprehensive application traffic analysis and reporting across the entire network.
Attempting to use Mikrotik's L7 firewall rules to generate the kind of application breakdown reports the user desires would be impractical due to its high resource consumption, limited inspection capabilities (especially with encrypted traffic), and the complexity of managing numerous patterns. For meaningful application-level visibility, relying on off-router solutions like ntopng with nDPI or ElastiFlow with its enrichment capabilities is essential. These tools are designed for this purpose and employ more sophisticated and efficient techniques.

6. Deployment Considerations for Your Home Network

Successfully implementing a network traffic monitoring solution involves considering the resources on both the Mikrotik router and the system designated for collection and analysis.

6.1 Minimizing Router CPU Load

The primary function of the Mikrotik router is to route traffic efficiently. While any monitoring task will add some overhead, the goal is to minimize this impact.

- **Traffic Flow (NetFlow/IPFIX Export):** This is generally the most router-friendly method for comprehensive data collection. The router's main tasks are to identify flows, count packets/bytes, and export these records periodically. The CPU load is typically low to moderate, depending on the number of active flows and the export frequency.[3]
- **Torch:** Can be CPU-intensive if run broadly without filters on a busy interface.[5] Best used for short, targeted investigations.
- **Packet Sniffer:** Full packet capture, especially without filters, can heavily load the CPU and I/O.[7] Streaming via TZSP with specific mangle rules is more targeted.
- **Layer 7 Firewall Inspection:** As discussed, this is extremely resource-intensive and should be avoided for general monitoring.[60]

Offloading the actual collection, processing, and analysis of flow data to a separate,

dedicated system is crucial for maintaining router performance and enabling the use of more powerful analysis tools.

6.2 Collector/Analyzer System Requirements (Server/Container/Desktop)

The system chosen to run the collector and analysis software will need adequate resources. The user's macOS desktop could run Docker containers, but an always-on Linux server (physical or virtual) is often preferred for continuous monitoring.

- **ntopng + netflow2ng:**
  - netflow2ng itself is lightweight, being written in Go.
  - ntopng's resource requirements depend on the traffic volume, number of active hosts, and data retention settings. For a typical home network, a system with 2-4 CPU cores and 4-8 GB of RAM should be sufficient. Redis, used by ntopng, also consumes memory based on the amount of data it's caching. This setup can often run comfortably in a Docker container or a modest Linux VM.
- **ElastiFlow + Elasticsearch/OpenSearch Stack:**
  - The ElastiFlow Unified Flow Collector itself is designed to be efficient. Documentation suggests it can handle high flow rates with minimal CPU.[33]
  - The Elasticsearch/OpenSearch backend is the more resource-demanding component. It requires significant RAM (ElastiFlow's quick start guide for a 500 FPS setup on Ubuntu suggests 5GB JVM heap for Elasticsearch on a 16GB RAM system, with a 10GB memory limit for the Elasticsearch container [34]). Disk I/O performance and sufficient disk space are also critical for Elasticsearch/OpenSearch. Running this entire stack via Docker is common and recommended by ElastiFlow.
- **Prometheus + Grafana + Flow Exporter:**
  - Prometheus and Grafana are generally efficient for their roles.
  - The resource load would primarily come from the flow exporter (e.g., netflow2ng in Prometheus mode, or pmacct). If pmacct is used with complex aggregation or if it's writing to an intermediate database that's then scraped, those components add to the load.
  - This stack is generally less demanding than a full ELK/OpenSearch setup if only aggregated metrics are stored.

The choice of solution should align with the available hardware for offloading. A Raspberry Pi 4 (with 4GB or 8GB RAM) could potentially run ntopng + netflow2ng for a home network, but would likely struggle with a full Elasticsearch stack. The user's existing server or container infrastructure will dictate the feasibility of more

resource-intensive options.

6.3 Data Storage Planning

Flow data, even though it's metadata, can accumulate rapidly, especially if detailed records are kept for extended periods.

- **ntopng:** Stores some real-time data in Redis. Historical flow data can be written to disk by ntopng itself, or for more advanced historical analysis, integration with databases like ClickHouse is possible (though ClickHouse integration is typically an enterprise feature).[16] The community version's long-term storage capabilities might be more basic. netflow2ng itself does not store data long-term; it forwards it to ntopng.
- **Elasticsearch/OpenSearch:** These are designed to handle large volumes of data. However, disk space is a primary consideration. Index Lifecycle Management (ILM) is crucial for managing data retention – automatically rolling over to new indices, moving older data to less expensive storage tiers (if applicable), and eventually deleting old data to prevent disk exhaustion.[39] ElastiFlow often provides default ILM policies.
- **Prometheus:** Stores time-series data, which is generally much more compact than individual flow records. Storage is configured within Prometheus itself, with retention periods defining how long metrics are kept.

The user should consider how long they wish to retain detailed flow data and plan disk space accordingly. For home use, retaining highly granular data for many months might be unnecessary and consume significant storage. Aggregated summaries might be more useful for long-term trending.

The resource allocation for the offloaded analysis system directly correlates with the complexity and depth of the desired analysis. Simpler setups like ntopng with netflow2ng are less demanding than a full ElastiFlow deployment with Elasticsearch, allowing the user to scale the solution to their available hardware.

7. Paid Options with Personal-Use Friendly Pricing (Brief Overview)

While the focus is on free and open-source solutions, some commercial products offer free tiers or pricing that might be considered "personal-use friendly."

7.1 PRTG Network Monitor

PRTG Network Monitor by Paessler is a well-known commercial monitoring tool.

- **Free Tier:** PRTG offers a freeware edition that allows the use of **up to 100 sensors for free, for life**.[63] This is a very generous offering for personal or home lab use. A "sensor" in PRTG typically monitors one aspect of a device or service (e.g., CPU load, interface traffic, a specific NetFlow data stream).
- **Flow Monitoring Capabilities:** PRTG supports NetFlow (v5, v9), IPFIX, and sFlow.[65] It has dedicated sensors for these flow types.
- **Features:** It can identify top talkers (by IP, application, protocol), top connections, and provide detailed traffic reports and visualizations.[66] Users can filter traffic by various criteria.
- **Mikrotik Support:** While not explicitly detailed for Mikrotik in the provided materials, PRTG aims for broad vendor compatibility.[65] Community discussions indicate users successfully employ PRTG with Mikrotik for NetFlow monitoring.[68] Setup involves configuring the Mikrotik router to send flow data to the PRTG server's IP address and the listening port configured in the PRTG NetFlow sensor (e.g., UDP 2055).
- **Application Visibility:** PRTG provides insights into application traffic, often based on protocol and port, and can show traffic from specific known applications like Citrix, FTP, etc., through its NetFlow sensor configurations.[65]

The 100 free sensors offered by PRTG could be a very attractive option if a polished, commercially supported product experience is desired without any cost for a typical home network. If a NetFlow source from the Mikrotik router (monitoring the LAN bridge, for instance) consumes only a few sensors, this free tier would be highly viable.

7.2 SolarWinds NetFlow Traffic Analyzer (NTA)

SolarWinds NTA is a powerful, enterprise-grade network traffic analysis tool.

- **Capabilities:** Supports NetFlow (v5, v9), IPFIX, sFlow, J-Flow, and Huawei NetStream.[70] It excels at identifying top talkers, applications, and protocols, and offers extensive reporting and alerting.
- **Pricing:** The pricing structure starts at $1,337 according to one source [70], with another listing a 1-year subscription for an SL100 license at $1,242.99.[73] This level of pricing is generally **not considered personal-use friendly** for a home network.
- **Mikrotik Support:** Given its broad vendor support for standard flow protocols, it would likely handle Mikrotik-exported flows correctly. Some ManageEngine (a competitor) documentation discusses Mikrotik NetFlow collection, suggesting industry tools generally support it.[74]

While SolarWinds NTA is a feature-rich product, its cost places it outside the typical budget for home network monitoring, unless a specific, heavily discounted personal license exists that was not identified in the research.

8. Conclusion: Choosing the Right Path for Your Network

Selecting the optimal network traffic monitoring solution for a Mikrotik RouterOS v7 setup depends on the desired level of detail, available resources for offloading analysis, and the time investment for setup and maintenance.

Recap of Recommended Solutions:

- **For Comprehensive Free/Open-Source Analysis with Good Application Identification:**
  - **ntopng + netflow2ng:** An excellent choice if NetFlow v9 is sufficient. netflow2ng makes ntopng's powerful nDPI-based application identification and interactive flow exploration accessible without the cost of nProbe. Setup is moderately complex.
  - **ElastiFlow Unified Flow Collector (Community or Basic Tier) + Elasticsearch/OpenSearch:** Offers robust IPFIX/NetFlow v9 collection, excellent enrichment capabilities, and powerful pre-built dashboards. The free tiers (500 FPS limit) are suitable for home use. Setup complexity is moderate to high due to the backend stack.
- **For Quick Real-Time Checks:**
  - **Mikrotik Torch:** Ideal for on-the-spot analysis of live traffic on a specific interface.
- **For Deep-Dive Packet Analysis:**
  - **Mikrotik Packet Sniffer / Mangle sniff-tzsp to Wireshark:** Essential for detailed forensic investigation of specific packets or flows once identified.
- **For a Polished Free Commercial Option (Limited Sensors):**
  - **PRTG Network Monitor:** The 100-sensor free license provides a user-friendly, feature-rich option for NetFlow/IPFIX analysis if the sensor count is sufficient for the home network.

Guidance on Selection:
The user's technical proficiency allows for any of these setups.

- If the priority is rich application identification and interactive flow browsing with minimal cost and moderate setup effort, **ntopng + netflow2ng** is a strong contender.
- If advanced data retention, powerful search, and pre-built, highly visual

dashboards are desired, and the user can manage an Elasticsearch/OpenSearch deployment, **ElastiFlow's free tiers** offer significant value.
- If a more metrics-driven approach is preferred, and the user is already familiar with **Prometheus and Grafana**, leveraging netflow2ng's Prometheus exporter or setting up pmacct with a suitable exporter can provide valuable trend data, though with less granular flow exploration.
- For users preferring a commercial, out-of-the-box solution with a generous free tier, **PRTG** is worth evaluating.

The following table provides a comparative overview of the recommended free/open-source flow analysis solutions:

Solution	**Primary Data Export from Mikrotik**	**Ease of Setup (Collector & Backend)**	**Collector/Backend Resource Needs**	**Application ID Quality (Free Tier)**	**Top Talker Visibility**	**Dashboarding**	**Key Free Tier Limitations**
ntopng + netflow2ng	NetFlow v9	Moderate	Low-Moderate	Good (via nDPI)	Excellent	Built-in Web UI	netflow2ng is NetFlow v9 only; ntopng community may have limits on some advanced features
ElastiFlow + ELK/OpenSearch	IPFIX / NetFlow v9	Moderate-Complex	Moderate-High	Good-Very Good (IEs, enrichment)	Excellent	Kibana / OpenSearch Dashboards	500 FPS limit; IE limits (Basic tier better than Community)

Prometheus + Grafana (via netflow2ng)	NetFlow v9	Moderate	Low-Moderate	Basic (depends on exporter)	Good (for metrics)	Grafana	Metrics-focused; less interactive flow detail; netflow2ng is NetFlow v9 only

Ultimately, the best approach will balance the depth of insight required with the resources available and the user's preference for specific tools or ecosystems. Given the user's technical capabilities, any of the open-source solutions are achievable, offering powerful ways to understand home network traffic in detail.

Works cited

1. splynx.com, accessed May 23, 2025, https://splynx.com/blog/system-configuration/understanding-netflow-protocol-in-mikrotik-relevance-after-the-end-of-support-of-the-ip-accounting-in-routeros-7/#:~:text=RouterOS%207%20and%20the%20IP,conjunction%20with%20other%20monitoring%20tools.
2. IP Accounting Guide - DataTill, accessed May 23, 2025, https://www.datatill.com/ip-accounting-guide/
3. Configuring NetFlow in MikroTik [Trisul Network Analytics Developer ..., accessed May 23, 2025, https://trisul.org/devzone/doku.php/netflow:mikrotik
4. MikroTik RouterOS - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/6.3/guides/device_flow_mikrotik_routeros/
5. A Beginner's Guide to Using MikroTik Torch for Network Traffic Analysis - j2sw Blog, accessed May 23, 2025, https://blog.j2sw.com/equipment-2/mikrotik/a-beginners-guide-to-using-mikrotik-torch-for-network-traffic-analysis/
6. Torch - RouterOS - MikroTik Documentation, accessed May 23, 2025, https://help.mikrotik.com/docs/display/ROS/Torch
7. Packet Sniffer - RouterOS - MikroTik Documentation, accessed May 23, 2025, https://help.mikrotik.com/docs/spaces/ROS/pages/8323088/Packet+Sniffer
8. Manual:Tools/Packet Sniffer - MikroTik Wiki, accessed May 23, 2025, https://wiki.mikrotik.com/Manual:Tools/Packet_Sniffer
9. Tips4Tiks – Packet Capture with MikroTik - Admiral Platform, accessed May 23, 2025, https://admiralplatform.com/tips4tiks-packet-capture-with-mikrotik/
10. Packet streaming from ROS and remote capture with Wireshark - MikroTik - Forum, accessed May 23, 2025, https://forum.mikrotik.com/viewtopic.php?t=182568

11. Howto capture traffic from a Mikrotik router on Linux - Robert Penz Blog, accessed May 23, 2025, https://robert.penz.name/737/howto-capture-traffic-from-a-mikrotik-router-on-linux/
12. Mangle - RouterOS - MikroTik Documentation, accessed May 23, 2025, https://help.mikrotik.com/docs/spaces/ROS/pages/48660587/Mangle
13. Profiler - RouterOS - MikroTik Documentation, accessed May 23, 2025, https://help.mikrotik.com/docs/spaces/ROS/pages/8323153/Profiler
14. ntopng – ntop, accessed May 23, 2025, https://www.ntop.org/products/traffic-analysis/ntop/
15. Ntopng Network Traffic Monitoring Tool - Google Docs, accessed May 23, 2025, https://docs.google.com/document/preview?hgd=1&id=1QPlNU5L2P6XdMQ8r53xnJVt2SXyJJKI5pLt7irKApk4
16. Traffic Dashboard — ntopng 6.5 documentation, accessed May 23, 2025, https://www.ntop.org/guides/ntopng/user_interface/network_interface/dashboard/dashboard.html
17. Available Versions & Licensing — ntopng 6.5 documentation, accessed May 23, 2025, https://www.ntop.org/guides/ntopng/versions_and_licensing.html
18. How to Analyse MikroTik Traffic Using ntopng – ntop, accessed May 23, 2025, https://www.ntop.org/ntopng/how-to-analyse-mikrotik-traffic-using-ntopng/
19. Using NetFlow with nProbe for ntopng | Weberblog.net, accessed May 23, 2025, https://weberblog.net/using-netflow-with-nprobe-for-ntopng/
20. nProbe – ntop, accessed May 23, 2025, https://www.ntop.org/products/netflow/nprobe/
21. FAQs - Ntop, accessed May 23, 2025, https://www.ntop.org/support/documentation/faq/
22. synfinatic/netflow2ng: NetFlow v9 collector for ntopng - GitHub, accessed May 23, 2025, https://github.com/synfinatic/netflow2ng
23. How to Install Ntopng on Ubuntu 20.04 | Vultr Docs, accessed May 23, 2025, https://docs.vultr.com/how-to-install-ntopng-on-ubuntu-20-04
24. Installation — ntopng 6.5 documentation, accessed May 23, 2025, https://www.ntop.org/guides/ntopng/installation.html
25. Docker build for ntopng + nDPI + Maxmind GeoIP + netflow2ng (FOSS nprobe alternative) - GitHub, accessed May 23, 2025, https://github.com/ThePlexus/ntopng-docker
26. Using ntopng Community Edition to Analyze LAN Traffic - Virtualization Review, accessed May 23, 2025, https://virtualizationreview.com/articles/2025/03/10/using-ntopng-community-edition-to-analyze-lan-traffic.aspx
27. Unified Flow Collector Introduction - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/6.4/docs/flowcoll/introduction/
28. robcowart/elastiflow: Network flow analytics (Netflow, sFlow ... - GitHub, accessed May 23, 2025, https://github.com/robcowart/elastiflow
29. ElastiFlow Documentation | ElastiFlow, accessed May 23, 2025, https://docs.elastiflow.com/

30. General Configuration - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/docs/flowcoll/config_gen/
31. Licensing - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/docs/config_ref/flowcoll/license
32. General Configuration - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/6.0/config_gen/
33. Kibana Dashboard ElastiFlow 4.0.1 issue - Elastic Discuss, accessed May 23, 2025, https://discuss.elastic.co/t/kibana-dashboard-elastiflow-4-0-1-issue/311797
34. All-in-One Quickstart for NetObserv Flow with Elasticsearch ..., accessed May 23, 2025, https://docs.elastiflow.com/docs/flowcoll/install_docker_ubuntu_elastic_stack/
35. Installation Guide for Debian, RedHat, and Docker - YouTube, accessed May 23, 2025, https://www.youtube.com/watch?v=707ReHCp7g0
36. elastiflow/flow-collector - Docker Image | Docker Hub, accessed May 23, 2025, https://hub.docker.com/r/elastiflow/flow-collector
37. RHEL/CentOS - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/6.4/docs/data_platforms/elastic/install_redhat/
38. Planning to Optimization: How ElastiFlow Simplifies Cloud Migration, accessed May 23, 2025, https://www.elastiflow.com/blog/posts/from-planning-to-optimization-how-elastiflow-simplifies-cloud-migration
39. ElastiFlow Tips and Tricks for Everyone, accessed May 23, 2025, https://www.elastiflow.com/blog/posts/elastiflow-tips-and-tricks-for-everyone
40. Optimize Your Network Performance with Advanced Network Management Software - ElastiFlow, accessed May 23, 2025, https://www.elastiflow.com/network-performance
41. Netflow collection and visualization with Elastiflow | Black Art of ..., accessed May 23, 2025, https://edennington.wordpress.com/2020/02/24/netflow-collection-and-visualization-with-elastiflow/
42. ElastiFlow and Rohde & Schwarz collaborate to deliver unmatched network traffic insights, accessed May 23, 2025, https://www.rohde-schwarz.com/sg/about/news-press/all-news/elastiflow-and-rohde-schwarz-collaborate-to-deliver-unmatched-network-traffic-insights-press-release-detailpage_229356-1562944.html
43. NetQuest and ElastiFlow Deliver 100G Encrypted Traffic Analysis for Threat Hunting at Telco Scale, accessed May 23, 2025, https://netquestcorp.com/netquest-and-elastiflow-deliver-100g-encrypted-traffic-analysis/
44. A comprehensive guide for workload migration to the cloud with the Search AI Platform, ElastiFlow, and Kyndryl | Elastic Blog, accessed May 23, 2025, https://www.elastic.co/blog/guide-workload-migration-cloud-elasticsearch-elastiflow-kyndryl
45. neptune-networks/flow-exporter: Export network flows from Kafka to Prometheus - GitHub, accessed May 23, 2025,

https://github.com/neptune-networks/flow-exporter
46. Network Flow Analysis With Prometheus - brooks.sh, accessed May 23, 2025, https://brooks.sh/2019/11/17/network-flow-analysis-with-prometheus/
47. Top 5 Telemetry and Observability Tools for Supporting SONiC Networks, accessed May 23, 2025, https://be-net.com/sonic-network-observability-and-monitoring/
48. Exporters and integrations - Prometheus, accessed May 23, 2025, https://prometheus.io/docs/instrumenting/exporters/
49. Writing exporters - Prometheus, accessed May 23, 2025, https://prometheus.io/docs/instrumenting/writing_exporters/
50. NetFlow - Learn Netdata, accessed May 23, 2025, https://learn.netdata.cloud/docs/collecting-metrics/networking-stack-and-network-interfaces/netflow
51. Monitoring Jaeger, accessed May 23, 2025, https://www.jaegertracing.io/docs/1.14/monitoring/
52. zaneclaes/network-traffic-metrics: Monitor network traffic with Prometheus & Grafana - GitHub, accessed May 23, 2025, https://github.com/zaneclaes/network-traffic-metrics
53. prometheus/node_exporter: Exporter for machine metrics - GitHub, accessed May 23, 2025, https://github.com/prometheus/node_exporter
54. pmacct/FAQS at master - GitHub, accessed May 23, 2025, https://github.com/pmacct/pmacct/blob/master/FAQS
55. pmacct project: IP accounting iconoclasm, accessed May 23, 2025, http://www.pmacct.net/
56. pmacct: introducing BGP na2vely into a NetFlow/sFlow collector - Netnod, accessed May 23, 2025, https://www.netnod.se/sites/default/files/2016-12/pmacct-plucente.pdf
57. What is nDPI - Ntop, accessed May 23, 2025, https://www.ntop.org/guides/nDPI/what_is_ndpi.html
58. Home · ntop/nDPI Wiki - GitHub, accessed May 23, 2025, https://github.com/ntop/nDPI/wiki
59. nDPI – ntop, accessed May 23, 2025, https://www.ntop.org/products/deep-packet-inspection/ndpi/
60. Layer7 - RouterOS - MikroTik Documentation, accessed May 23, 2025, https://help.mikrotik.com/docs/spaces/ROS/pages/130220161/Layer7
61. Configuring MikroTik Firewall - Best Practices - Tech@Layer-x, accessed May 23, 2025, https://tech.layer-x.com/configuring-mikrotik-firewall-best-practices/
62. Controlling Network Traffic using MikroTik RouterOS, accessed May 23, 2025, https://mum.mikrotik.com/presentations/IT14/touw.pdf
63. PRTG - Monitoring on the App Store - Apple, accessed May 23, 2025, https://apps.apple.com/us/app/prtg-monitoring/id326306472
64. PRTG Manual: Available Licenses - Paessler, accessed May 23, 2025, https://www.paessler.com/manuals/prtg/available_licenses
65. NetFlow Monitoring | PRTG - Paessler, accessed May 23, 2025, https://www.paessler.com/netflow_monitoring

66. NetFlow Monitoring | PRTG - Paessler, accessed May 23, 2025, https://www.paessler.com/monitoring/technology/netflow-monitoring-tool
67. IPFIX Monitoring | PRTG - Paessler, accessed May 23, 2025, https://www.paessler.com/monitoring/technology/ipfix-monitoring
68. Setting up PRTG : r/prtg - Reddit, accessed May 23, 2025, https://www.reddit.com/r/prtg/comments/18zxefs/setting_up_prtg/
69. Trying to setup netflow from Mikrotik to a PRTG server at a different site. - Reddit, accessed May 23, 2025, https://www.reddit.com/r/mikrotik/comments/41ldit/trying_to_setup_netflow_from_mikrotik_to_a_prtg/
70. NetFlow Traffic Analyzer | Use Cases - SolarWinds, accessed May 23, 2025, https://www.solarwinds.com/netflow-traffic-analyzer/use-cases
71. NetFlow Traffic Analyzer Product Overview Video - SolarWinds, accessed May 23, 2025, https://www.solarwinds.com/resources/video/netflow-traffic-analyzer-overview
72. Network Traffic Analysis Tool - Analyze Your Network - SolarWinds, accessed May 23, 2025, https://www.solarwinds.com/netflow-traffic-analyzer/use-cases/network-traffic-analysis
73. SolarWinds NetFlow Traffic Analyzer for SolarWinds Network Performance Monitor SL100 - subscription license (1 year) - Insight, accessed May 23, 2025, https://www.insight.com/en_US/shop/product/100061/solarwinds/100061/SolarWinds-NetFlow-Traffic-Analyzer-for-SolarWinds-Network-Performance-Monitor-SL100-subscription-license-1-year-1-license/
74. Mikrotik & Netflow Analyzer - ManageEngine Pitstop, accessed May 23, 2025, https://pitstop.manageengine.com/portal/en/community/topic/mikrotik-netflow-analyzer
75. Licensing - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/6.1/config_ref_license/
76. Licensing - ElastiFlow Documentation, accessed May 23, 2025, https://docs.elastiflow.com/5.6/config_ref_license/