# Technical Report — Network Tool for Security Edge Inc.

## 1. Overview

This document provides a technical description of the *Network Tool* developed for **Security Edge Inc.**, intended for use by MSSP (Managed Security Service Providers) team.
The tool provides essential network diagnostic and reconnaissance capabilities in a lightweight, cross-platform script written in Python 3.

It is designed to:

- Identify and monitor active devices on a network;
- Test connectivity to hosts;
- Scan open TCP ports;
- Measure network bandwidth usage;
- Detect high-traffic anomalies.

All operations are performed locally using standard system utilities and Python's built-in libraries, ensuring compatibility with **Windows, macOS, and Linux**.

## 2. Architecture

The tool follows a modular, object-oriented architecture centered on the `NetworkTool` class, providing isolated methods for each feature:

| Feature | Method | Description |
|---|---|---|
| Host Ping | `ping_host()` | Performs ICMP ping using system-native commands (`ping -n` / `ping -c`). |
| Port Scan | `scan_ports()` | Concurrent TCP connect-scan using Python sockets and `ThreadPoolExecutor`. |

| | | |
|---|---|---|
| Network Traffic | `get_network_traffic()` | Measures bytes sent/received over a time interval (requires `psutil`). |
| Device Discovery | `discover_devices()` | Runs `arp -a`, parses output, and lists active IP/MAC pairs. |
| Alert System | `alert()` | Prints highlighted warning messages for anomalies. |
| Interactive Menu | `run()` | Provides a user-friendly text interface for all functionalities. |

---

# 3. Cross-Platform Compatibility

- **Ping:** Automatically selects parameters based on OS (`-n` for Windows, `-c` for UNIX).
- **ARP:** Uses `arp -a`, compatible across Windows and UNIX-based systems.
- **Colors:** ANSI colors are disabled on Windows if unsupported.
- **Dependencies:** Only one optional external dependency (`psutil`), gracefully handled if missing.

# 4. Functional Workflow

1. **User Menu:**
   The program displays an interactive menu where the user selects an operation (ping, scan, traffic, or discovery).

2. **Input Handling:**
   All user inputs (host, port range, duration) are validated to prevent crashes and handle invalid data gracefully.

3. **Execution:**
   - **Ping:** Executes OS command to verify host availability.
   - **Port Scan:** Resolves host IP and scans ports concurrently for performance.
   - **Traffic Monitor:** Captures I/O deltas using `psutil`.
   - **ARP Discovery:** Parses ARP table with regex patterns for both OS types.

4. **Output:**
   Results are displayed in color-coded, human-readable format.
   Alerts are triggered if anomalies (e.g., host down or high traffic) are detected.

# 5. Security Considerations

- The tool operates with **read-only system commands**, avoiding intrusive operations.
- It does **not** perform raw packet injection or privilege escalation.
- ARP discovery may require elevated privileges depending on the OS.
- Recommended for use in **authorized internal networks only**.

# 6. Example Output

=== SECURITY EDGE INC NT ===
1) Ping a host
2) Scan open TCP ports
3) Measure network traffic (requires psutil)
4) Discover devices (ARP - may require privileges)
5) Exit

Choice (1-5): 1
Host or IP to ping: 8.8.8.8
Pinging 8.8.8.8 1 time(s)...
8.8.8.8 is UP

---

# 7. Dependencies

| Library | Purpose | Installation |
|---|---|---|
| `psutil` | Optional: network I/O measurement | `pip install psutil` |

All other functionalities use built-in Python libraries (`socket`, `subprocess`, `threading`, `re`, etc.).

# 8. Known Limitations

- Requires network interface access and ARP cache availability.

- Port scans rely on TCP connect tests (no UDP support).
- Real-time traffic metrics are limited by the refresh interval.

# 9. Future Improvements

- Add logging and report generation (CSV/JSON).
- Integrate SNMP or Nmap APIs for deeper network insight.
- Provide GUI (Tkinter or web interface).
- Add remote alerting (email/webhook integration).

# 10. Conclusion

The **Security Edge Network Tool** provides a robust and lightweight solution for real-time network visibility.
It is portable, auditable, and safe for internal security assessments, offering MSSP teams a practical instrument for day-to-day network operations.