



SYNAPSE
SONAR

Agent Setup & Configuration Guide

Deploying collectors that feed Synapse Sonar

Version MVP 0.1 · 30 June 2026

Synapse IronNode product family

1. Overview

Synapse Sonar is agentless at its core: it **receives** network flow data pushed to it by collectors you deploy. This guide covers the three collectors and the NetscanXi vulnerability connector.

Collector	Runs on	Sees	Use when
Linux host agent	Each Debian/Ubuntu host	That host's own connections	You can install software on the host
Windows host agent	Each Windows host	That host's own TCP connections	You can install software on the host
Passive sensor	A host on a SPAN/mirror port	All mirrored traffic	The device cannot take an agent

The Linux agent and sensor are pure Python 3 standard library; the Windows agent uses built-in PowerShell. No third-party packages are required on the endpoints.

2. Prerequisites

- A running Synapse Sonar instance reachable from your hosts (e.g. <https://sonar:3000>).
- The **NetscanXi integration enabled** in Synapse Sonar — this provides the ingest key.
- Network egress from each host to the Sonar address/port.

3. Get the ingest key

All collectors authenticate with a single **ingest key**:

- Sign in as a Tenant Admin and open **Integrations**.
- Toggle **NetscanXi** on and click **Generate API key**.
- Copy the key immediately — it is shown only once. Regenerating revokes the old key.

Note. The same key is used by the host agents, the passive sensor, and the NetscanXi vulnerability feed.

4. Download from the Agents tab (recommended)

The fastest path is the built-in **Agents & Collectors** tab (Tenant Admin only):

- Pick a platform, confirm the **Sonar address** and paste/generate the **ingest key**.
- Click **Download bundle (.zip)**. The zip contains the agent, installer, your pre-filled config, and a QUICKSTART.

Or download the raw files directly from `/agent.s/` on the server.

5. Linux host agent (Debian / Ubuntu)

Reports the host's own established connections via ss. Runs unprivileged as a systemd service.

```
unzip synapse-sonar-linux-agent.zip -d sonar-agent
cd sonar-agent
sudo ./install.sh
sudo cp sonar-agent.env /etc/sonar-agent/sonar-agent.env
sudo systemctl enable --now sonar-agent
journalctl -u sonar-agent -f          # watch it push flows
```

If the installer is not executable after transfer, run `sudo bash install.sh` instead.

Configuration (/etc/sonar-agent/sonar-agent.env)

Key	Meaning	Default
SONAR_URL	Synapse Sonar base URL	—
SONAR_INGEST_KEY	Ingest key (Bearer)	—
SONAR_INTERVAL	Seconds between pushes	30
SONAR_VERIFY_TLS	Verify TLS cert (false for self-signed)	true

6. Windows host agent

Reports the host's own TCP connections via `Get-NetTCPConnection`. Installs as a `SYSTEM` scheduled task that starts at boot. From an **elevated** PowerShell:

```
Expand-Archive synapse-sonar-windows-agent.zip -DestinationPath sonar-agent
cd sonar-agent
powershell -ExecutionPolicy Bypass -File .\install.ps1
copy config.json "$env:ProgramData\SonarAgent\config.json"
Restart-ScheduledTask -TaskName SynapseSonarAgent
```

Configuration (%ProgramData%\SonarAgent\config.json)

Key	Meaning	Default
SonarUrl	Synapse Sonar base URL	—
IngestKey	Ingest key (Bearer)	—
IntervalSeconds	Seconds between pushes	30
VerifyTls	Verify TLS cert	true

Logs: `%ProgramData%\SonarAgent\sonar-agent.log`. Remove with `Unregister-ScheduledTask -TaskName SynapseSonarAgent -Confirm:$false`.

7. Passive sensor (agentless devices)

For devices that cannot run an agent (appliances, IoT, printers, OT). Deploy on a Linux host attached to a switch **SPAN / mirror port** (or inline tap / gateway) so it can observe traffic to and from those devices. It sniffs IPv4 TCP/UDP, aggregates flows with byte counts, and excludes its own traffic to Sonar.

```
unzip synapse-sonar-sensor-agent.zip -d sonar-sensor
cd sonar-sensor
sudo ./install.sh
sudo cp sonar-sensor.env /etc/sonar-sensor/sonar-sensor.env
sudo systemctl enable --now sonar-sensor
journalctl -u sonar-sensor -f
```

Docker alternative (host networking is required to see the mirrored interface):

```
docker run -d --name sonar-sensor --network host \
  --cap-add NET_RAW --cap-add NET_ADMIN \
  -e SONAR_URL=https://sonar:3000 -e SONAR_INGEST_KEY=nsx_sensor_xxxx \
  -e SONAR_IFACE=eth1 \
  -v "$PWD/sonar-sensor.py:/app/sonar-sensor.py:ro" \
  python:3.12-alpine python /app/sonar-sensor.py
```

Configuration (/etc/sonar-sensor/sonar-sensor.env)

Key	Meaning	Default
SONAR_URL	Synapse Sonar base URL	—
SONAR_INGEST_KEY	Ingest key (Bearer)	—
SONAR_IFACE	Interface on the mirror port	(all)
SONAR_FLUSH	Aggregation window (seconds)	30
SONAR_VERIFY_TLS	Verify TLS cert	true

Sizing. The sensor only sees what the mirror port forwards. Size the SPAN session and the sensor NIC for the mirrored volume, or flows will be dropped upstream.

8. NetscanXi vulnerability feed

NetscanXi pushes scan results to Synapse Sonar (push model). In the NetscanXi card, copy the **Endpoint URL** and **API key**, then configure your scanner to POST findings on each scan. Expected payload:

```
POST /api/ingest/vulnerabilities
Authorization: Bearer <ingest-key>

{ "findings": [
  { "ip": "10.0.3.30", "cveId": "CVE-2024-1234",
    "cvssScore": 9.8, "severity": "CRITICAL",
    "title": "PostgreSQL RCE" } ] }
```

If the scanner cannot post in this format, run the connector script (netscan-connector) on a schedule to translate its export and forward it. Findings are matched to assets by IP and upserted by CVE.

9. Verifying data flow

- Agent/sensor: `systemctl status` should show **active (running)**, and the logs should report pushed N flows.
- In Synapse Sonar, hosts appear as nodes on the **Topology Map** within a minute or two.
- Vulnerabilities appear once the NetscanXi feed sends data (and the toggle is on).

10. Troubleshooting

Symptom	Likely cause & fix
HTTP 401 from ingest	Wrong/expired ingest key, or NetscanXi integration disabled.
Service won't start (Linux)	Check SONAR_URL/SONAR_INGEST_KEY are set; review journalctl.
TLS errors	Self-signed cert — set verify-TLS to false, or install a trusted cert.
No flows from sensor	Confirm the mirror port forwards traffic and SONAR_IFACE is correct.
Nodes but no vulns	Enable NetscanXi and confirm findings include the host IP.