Kusanagi Kajiki

ICS/SCADA Passive Network Discovery
& Security Analysis Tool

A modern rewrite of NSA's GRASSMARLIN

Version 1.0 | February 2026

David — The Security Lead

Table of Contents

Chapter 1: Introduction

Chapter 2: Installation

Chapter 3: Quick Start Guide

Chapter 4: Core Features

- 4.1 PCAP Import & Live Capture
- 4.2 Protocol Detection
- 4.3 Topology Views
- 4.4 Deep Protocol Analysis
- 4.5 Device Identification & Enrichment
- 4.6 Signature Engine

Chapter 5: Security Analysis

- 5.1 MITRE ATT&CK; for ICS Detection
- 5.2 Purdue Model Analysis
- 5.3 Anomaly Detection
- 5.4 Baseline Drift

Chapter 6: External Tool Integration

Chapter 7: Physical Topology

Chapter 8: Reporting & Export

Chapter 9: Session Management

Chapter 10: Settings & Customization

Chapter 11: CLI Reference

Appendix A: Appendix A: Protocol Reference

Appendix B: Appendix B: ATT&CK; Technique Reference

Appendix C: Appendix C: Troubleshooting

Chapter 1: Introduction

Kusanagi Kajiki is a modern, ground-up rewrite of the NSA's **GRASSMARLIN** (archived 2023) — an ICS/SCADA passive network discovery and topology visualization tool. Built with a Rust backend (Tauri 2.0) and a SvelteKit (Svelte 5) frontend, it passively discovers and maps Industrial Control System devices by analyzing network traffic. The name is a bilingual nod to the original: ■ (kusa/grass) + marlin (kajiki/■■■), with Kusanagi (■■) referencing the legendary sword from Japanese mythology.

Why Passive-Only?

In operational technology environments, **active scanning can crash PLCs and disrupt physical processes**. Unlike IT networks where a port scan is routine, OT networks often contain devices running 20-year-old firmware that will fault or reboot when they receive unexpected packets. Kusanagi Kajiki operates in strictly passive mode — it analyzes captured traffic or integrates data from other tools, but **never generates a single packet on the production network**. Live capture uses promiscuous receive only.

Beyond GRASSMARLIN

While Kusanagi Kajiki implements every major GRASSMARLIN 3.2 feature (multi-PCAP import, logical and physical topology views, signature-based device identification, Cisco config import, connection tree), it extends far beyond the original with capabilities GRASSMARLIN never offered:

- MITRE ATT&CK; for ICS detection — automated mapping of observed behaviors to ICS attack techniques
- Purdue Model enforcement — auto-assign levels, detect cross-zone violations
- External tool integration — import Zeek logs, Suricata EVE JSON, Nmap/Masscan results
- Professional PDF assessment reports with executive summary and findings
- SBOM export aligned with CISA BOD 23-01 for federal compliance
- STIX 2.1 threat intelligence bundles
- Baseline drift detection — compare assessments over time with quantified scoring
- Modern dark/light theme, CLI support, and timeline scrubber for topology playback

Target Audience

Kusanagi Kajiki is built for **OT security assessors**, ICS incident responders, and compliance auditors who need safe, comprehensive visibility into industrial networks. It is designed to support the full assessment workflow: import captures, discover assets, identify protocols and vendors, flag security concerns, and produce deliverable reports.

Chapter 2: Installation

System Requirements

- Operating System: Linux (Fedora/RHEL, Ubuntu/Debian), macOS, or Windows 10/11
- RAM: 4 GB minimum, 8 GB recommended for large captures
- Disk: 500 MB for application + space for capture data and SQLite database
- Rust >= 1.77 (install via `rustup.rs`)
- Node.js >= 22 (install via nvm or official installer)
- libpcap development headers (platform-specific)

Fedora / RHEL (Primary Platform)

Install system dependencies, clone the repository, and build:

```
# Install system dependencies
sudo dnf install libpcap-devel webkit2gtk4.1-devel \
libsoup3-devel javascriptcoregtk4.1-devel

# Clone and build
git clone https://github.com/TheSecurityLead/KusanagiNoKajiki.git
cd KusanagiNoKajiki
npm install --legacy-peer-deps
npm run build

# Development mode (hot-reload)
npm run tauri dev

# Production build
npm run tauri build
```

Note: `--legacy-peer-deps` is required on Fedora due to a vite/svelte peer dependency conflict.

Ubuntu / Debian

```
sudo apt install libpcap-dev libwebkit2gtk-4.1-dev \
libappindicator3-dev librsvg2-dev patchelf

git clone https://github.com/TheSecurityLead/KusanagiNoKajiki.git
cd KusanagiNoKajiki
npm install
npm run build
npm run tauri dev
```

macOS

```
brew install libpcap
xcode-select --install
```

```
git clone https://github.com/TheSecurityLead/KusanagiNoKajiki.git
cd KusanagiNoKajiki
npm install
npm run build
npm run tauri dev
```

Windows

1. Install **Npcap** from npcap.com — check "Install Npcap in WinPcap API-compatible Mode".
2. Download the **Npcap SDK** and add `Lib/x64` to your `LIB` environment variable.

```
git clone https://github.com/TheSecurityLead/KusanagiNoKajiki.git
cd KusanagiNoKajiki
npm install
npm run build
npm run tauri dev
```

Live Capture Without Root (Linux)

To perform live packet capture without running as root, grant the binary the necessary capabilities:

```
sudo setcap cap_net_raw,cap_net_admin=eip \
src-tauri/target/release/kusanaginokajiki
```

Verify Installation

Run the test suite to confirm everything is working:

```
cd src-tauri && cargo test --all
# Expected: 127 tests passing

cargo clippy --all -- -D warnings
# Expected: zero warnings
```

Chapter 3: Quick Start Guide

This walkthrough covers the primary assessment workflow from PCAP import through report generation. Public ICS PCAP samples for testing are available from github.com/automayt/ICS-pcap and wiki.wireshark.org/SampleCaptures.

1. Launch the Application

Run `npm run tauri dev` for development mode, or execute the compiled binary. The application opens with the Capture view.

2. Import PCAP File(s)

Click **Import PCAP File(s)** in the Capture tab. The OS file picker appears — select one or multiple .pcap or .pcapng files. Each packet tracks which file it originated from. Import progress is displayed, and results appear immediately.

3. View the Connection Tree

The left panel shows the **Connection Tree**: expandable nodes listing every discovered IP address, their connections, and individual packet summaries (capped at 1,000 per connection). Right-click any connection for context menu options including View Frames and Open in Wireshark.

4. Explore the Logical Topology

Switch to the **Topology** tab. The graph renders using an fcose (force-directed compound spring embedder) layout, with nodes grouped by subnet. Right-click nodes to watch neighbors, create filtered views, show in physical topology, or open in Wireshark. Use the timeline scrubber at the bottom to replay topology construction chronologically.

5. Inspect Devices

Open the **Inventory** tab to see all discovered assets in a sortable, searchable table. Each asset shows IP, MAC, vendor (from OUI), protocols, device type, confidence score, country flag (for public IPs), and Purdue level. Click any asset to open the detail panel showing Modbus/DNP3 function code analysis, register ranges, and polling intervals.

6. Run Security Analysis

Navigate to the **Analysis** tab and click **Run Analysis**. The engine automatically maps behaviors to MITRE ATT&CK; for ICS techniques, assigns Purdue Model levels, and scores anomalies. Results appear across four sub-tabs: Summary, Findings, Purdue, and Anomalies.

7. Review Findings

The Findings sub-tab shows all detected issues sorted by severity (Critical, High, Medium, Low, Info). Each finding includes the ATT&CK; technique ID, affected assets, a description, and recommended mitigations. Click any asset link to navigate to it in the Inventory.

8. Export a Report

Open the **Export** tab. Choose your export format: PDF (professional assessment report), CSV/JSON (raw data), SBOM (CISA BOD 23-01 asset inventory), or STIX 2.1 (threat intelligence bundle). For PDF reports, fill in assessor name, client name, and date, then click Generate.

Chapter 4: Core Features

4.1 PCAP Import & Live Capture

Kusanagi Kajiki supports both offline PCAP analysis and real-time live capture.

PCAP Import

- Multi-file import — select multiple PCAP/PCAPNG files simultaneously
- Per-packet origin tracking — every packet records which file it came from
- Connection deduplication — bidirectional flows merged with packet and byte counts
- Immediate topology rendering after import completes

Live Capture

- Interface selection from all available network interfaces
- Start, pause, resume, and stop controls
- BPF filter support (e.g., `tcp port 502`)
- Ring buffer holds up to 1,000,000 packets for PCAP save on stop
- Real-time streaming — topology and asset views update every 500ms during capture
- Save captured packets to a PCAP file at any time

4.2 Protocol Detection

The parser identifies 19 protocols across OT and IT networks using a multi-layered approach: port-based matching, YAML signature pattern matching, and deep packet inspection.

Protocol	Port(s)	Category	Detection
Modbus TCP	502	OT	Deep parse
DNP3	20000	OT	Deep parse
EtherNet/IP (CIP)	44818, 2222	OT	Signature
BACnet/IP	47808	OT	Signature
S7comm	102	OT	Signature
OPC UA	4840	OT	Port + Signature
IEC 60870-5-104	2404	OT	Port
PROFINET	34962-34964	OT	Port
MQTT	1883, 8883	OT/IIoT	Port
HART-IP	5094	OT	Port

Protocol	Port(s)	Category	Detection
FF HSE	1089-1091	OT	Port
GE SRTP	18245-18246	OT	Port + Signature
Wonderware SuiteLink	5007	OT	Port + Signature
HTTP/HTTPS	80, 443	IT	Port
DNS	53	IT	Port
SSH	22	IT	Port
RDP	3389	IT	Port
SNMP	161, 162	IT	Port
Telnet	23	IT	Port

Detection depth: *Port* = identified by TCP/UDP port number. *Signature* = matched by YAML payload or OUI patterns. *Deep parse* = full protocol dissection with function code analysis, device identification, and behavioral profiling.

4.3 Topology Views

Logical View

The primary topology view uses a force-directed compound spring embedder (fcose) layout. Nodes are grouped by subnet using compound (parent) nodes. Edges are colored by protocol. Features include:

- Right-click context menu: watch neighbors, create filtered views, group by attribute
- Dynamic grouping — regroup nodes by subnet, protocol, vendor, or device type
- Filtered views — create multiple simultaneous sub-topology tabs
- Watch tabs — N-degree neighborhood (1-5 hops) with auto-refresh
- Drift highlighting — new and changed nodes highlighted after baseline comparison

Physical View

Displays Cisco switch/port topology from imported IOS configurations. Switches render as compound nodes with physical port child nodes. CDP link edges connect switches. Cross-reference with the logical view: right-click any asset in the logical view and select "Show in Physical" to highlight the corresponding switch port.

Mesh View

An all-to-all connection matrix showing every pair of communicating devices. Supports protocol and time-range filters.

Timeline Scrubber

A playback bar at the bottom of the logical view. Drag the slider or press play to watch the topology build chronologically as packets were captured. Adjustable playback speed.

4.4 Deep Protocol Analysis

Modbus Deep Parse

- MBAP header parsing (transaction ID, protocol ID, length, unit ID)
- Function code extraction and read/write classification
- Register range tracking (start address + quantity for each request)
- Master/slave role detection based on port direction and function codes
- FC 43 sub-function 14 (Read Device Identification) — extracts vendor name, product code, and revision directly from device responses, yielding confidence level 5
- FC 8 diagnostics detection — flags potential denial-of-service activity
- Polling interval computation from timestamp analysis across repeated requests

DNP3 Deep Parse

- Start byte validation (0x05 0x64) — rejects non-DNP3 traffic cleanly
- Function code extraction with master vs. outstation classification
- DNP3 source and destination address extraction from the link layer
- Unsolicited response detection (FC 130) — flags potential alarm manipulation
- Application layer function code analysis

4.5 Device Identification & Enrichment

Kusanagi Kajiki uses multiple techniques to identify and enrich discovered devices:

YAML Signature Engine

25 YAML signature files covering 13 OT protocols and vendor-specific patterns. Signatures match on combinations of port, MAC OUI prefix, and payload byte patterns. Each match produces a vendor label, device type, and confidence score.

Confidence Scoring (1-5)

Level	Source	Example
1	Port match	Traffic on port 502 → probably Modbus
2	Pattern match	YAML signature payload pattern matched
3	OUI vendor	MAC prefix 00:0E:8C → Siemens AG
4	Payload analysis	Protocol-specific byte sequence in payload
5	Deep parse	FC 43/14 Device ID response → exact vendor/product

Additional Enrichment

- MAC OUI vendor lookup — IEEE OUI database with ~30,000 entries bundled
- GeoIP country lookup — DB-IP Lite database for public IP addresses, displayed as country flags

- Deep parse enrichment — Modbus FC 43/14 Device ID responses override vendor/product at confidence 5

4.6 Signature Engine

The built-in signature editor provides a CodeMirror 6 YAML editing environment with syntax highlighting. You can:

- Browse all 25 loaded signatures with metadata
- Edit signature YAML with syntax highlighting and validation
- Test a signature against currently loaded packet data
- Hot-reload all signatures from disk without restarting
- Create custom signatures for vendor-specific device identification

Signature YAML Format

```
name: modbus_schneider_m340
description: Schneider Electric Modicon M340
protocol: modbus
vendor: Schneider Electric
device_type: PLC
product_family: Modicon M340
confidence: 4
filters:
dst_port: 502
payload_contains: "\x00\x00\x00"
```

Chapter 5: Security Analysis

The Analysis tab provides automated security assessment of discovered OT networks. Click **Run Analysis** to execute all three analysis modules: ATT&CK; detection, Purdue Model assignment, and anomaly scoring.

5.1 MITRE ATT&CK; for ICS Detection

Kusanagi Kajiki maps observed network behaviors to MITRE ATT&CK; for ICS techniques. The following detections are implemented:

Detection	Technique	Severity	Trigger Condition
Broadcast writes	T0855	Critical	Modbus FC 5/6/15/16 to unit ID 0 or 255
High fan-out writes	T0855	High	Single source writing to 5+ target devices
Diagnostic abuse	T0814	High	FC 8 (Diagnostics) from non-engineering workstation
Reconnaissance	T0846	High	Unknown/IT device polling 3 or more PLCs
Unsolicited response	T0856	Medium	DNP3 FC 130 sent to an unknown master
Cross-zone traffic	T0886	Medium	Direct communication between Purdue L1 and L4

Findings appear in the **Findings** sub-tab, sorted by severity. Each finding includes the ATT&CK; technique ID, affected source and destination IPs, a human-readable description, and the confidence level of the detection. Click any IP address to navigate directly to that asset in the Inventory view.

5.2 Purdue Model Analysis

The Purdue Model (ISA-95 / IEC 62443) defines network segmentation zones for industrial environments. Kusanagi Kajiki automatically assigns Purdue levels based on observed behavior:

Level	Assignment	Criteria
L1 — Basic Control	PLCs, RTUs	Device serves OT ports (502, 20000, etc.)
L2 — Supervisory	HMIs	Device is a client connecting to multiple OT servers
L3 — Operations	Historians, SCADA	Historian, OPC UA, or high fan-out patterns
L4 — Enterprise	IT systems	Only IT protocols observed, no OT traffic

Manually assigned Purdue levels (set in the Inventory view) are preserved as **Manual** assignments and take priority over auto-assignment. Cross-zone violations (e.g., L1 device communicating directly with L4) are flagged as T0886 findings with severity based on the zone distance.

5.3 Anomaly Detection

Three categories of anomalies are scored:

Polling Deviation

For Modbus and DNP3 connections with regular polling patterns, the coefficient of variation (CV) of polling intervals is computed. A CV above 0.5 (50%) flags the connection as an anomaly; above 1.0 (100%) generates a formal finding.

Role Reversal

Detects when a device classified as a slave/outstation sends function codes that are typically only sent by masters (e.g., write commands originating from a PLC).

Unexpected Public IPs

Any public (non-RFC1918) IP address communicating via OT protocols on an internal network is flagged as Critical. This may indicate a misconfigured NAT, a compromised device phoning home, or an internet-exposed OT system.

5.4 Baseline Drift Detection

Baseline drift allows you to compare the current assessment against a previously saved session. This is essential for periodic assessments where you need to track network changes over time.

Workflow:

- Save the current assessment as a session (Capture tab → Save Session)
- Import new capture data (weeks/months later)
- Navigate to Analysis → Baseline Drift tab
- Select the previous session as the baseline
- Click Compare — the tool computes a quantified drift score

Results show: **new assets** (not in baseline), **missing assets** (in baseline but gone), and **changed assets** (different vendor, hostname, Purdue level, or protocols). The logical topology view highlights new nodes in green and changed nodes in amber.

Chapter 6: External Tool Integration

Kusanagi Kajiki integrates with common security tools to augment passive PCAP analysis. All imports merge into the same asset and topology pipeline as native PCAP data.

Zeek Log Import

Import Zeek TSV log files from the Capture tab. Supported log types:

- `conn.log` — connection metadata (IPs, ports, protocols, bytes)
- `modbus.log` — Modbus function codes and registers
- `dnp3.log` — DNP3 function codes and objects
- `s7comm.log` — S7comm protocol data

Zeek's `#fields` header is parsed automatically. The `#path` directive determines the log type. Missing fields (represented as `-`) are handled gracefully.

Suricata EVE JSON Import

Import Suricata's line-delimited EVE JSON output. Two event types are processed:

- **flow** events — extracted as connections with IP addresses, ports, and byte counts
- **alert** events — extracted with signature ID, severity, and description

Nmap XML Import

Import Nmap XML scan results. Discovered hosts, open ports, and service information are merged into the asset inventory. Data from Nmap is tagged with `[Nmap]` and `[active-scan]` to clearly distinguish it from passive observation.

Masscan JSON Import

Import Masscan JSON output. Like Nmap results, Masscan data is tagged with `[Masscan]` and `[active-scan]`. The importer handles Masscan's trailing comma and "finished" sentinel in the JSON output.

Wireshark Integration

Kusanagi Kajiki auto-detects Wireshark installations in standard system paths. When available:

- Right-click any node in the logical view → **Open in Wireshark** (filters by IP)
- Right-click any connection in the connection tree → **Open in Wireshark**
- **View Frames** — inspect individual packets in a table with timestamp, source, destination, protocol, length, and origin file

- **Export CSV** — export frame data to a CSV file via the save dialog

Chapter 7: Physical Topology

The Physical View maps devices to physical switch ports by importing Cisco IOS command output. This bridges the gap between logical network topology ("which IPs talk to each other") and physical infrastructure ("which device is plugged into which port").

Importing Cisco IOS Data

Four types of Cisco output can be imported from the Physical View panel:

Running Config

Paste or import the output of `show running-config`. The parser extracts: hostname, IOS version, interface definitions (descriptions, VLAN assignments, IP addresses, shutdown state, speed/duplex, switchport mode), VLAN definitions, and management IP.

MAC Address Table

Import `show mac address-table` output. Associates MAC addresses with physical switch ports and VLANs. Supports both Cisco dot format (0000.1111.2222) and standard colon/dash formats.

CDP Neighbors

Import `show cdp neighbors detail` output. Extracts device ID, IP address, platform, capabilities, and local/remote port mappings to draw switch-to-switch links.

ARP Table

Import `show arp` or `show ip arp` output. Maps IP addresses to MAC addresses, which are then correlated to physical switch ports via the MAC address table.

Physical Topology Graph

After importing switch data, the Physical View renders a Cytoscape graph with switches as compound nodes and physical ports as child nodes. CDP neighbor links appear as edges between switches. The detail panel shows port-level information when a port is selected.

Cross-Reference to Logical View

Right-click any asset in the **Logical View** and select **Show in Physical**. The Physical View opens with the corresponding switch ports highlighted, showing you exactly where a device is physically connected.

Chapter 8: Reporting & Export

The Export tab provides multiple output formats for assessment deliverables and data sharing.

PDF Assessment Report

Generate a professional PDF report containing:

- Executive summary with asset count, protocol breakdown, and key findings
- Asset inventory table with vendor, protocol, and confidence details
- Protocol analysis with traffic distribution
- Security findings table sorted by severity
- Recommendations based on detected issues

Configurable fields: assessor name, client/organization name, and assessment date. The PDF is generated using the genpdf Rust crate.

CSV Export

Export assets or connections as CSV files. Includes all fields visible in the Inventory and Connection Tree views. Useful for importing into spreadsheets or SIEM platforms.

JSON Export

Export the full topology (assets, connections, and metadata) or just assets as structured JSON. Suitable for programmatic analysis or integration with other tools.

SBOM — CISA BOD 23-01

Generate a Software Bill of Materials aligned with CISA Binding Operational Directive 23-01, which requires federal agencies to maintain inventories of networked assets. The SBOM lists all discovered OT devices with vendor, product, IP address, MAC address, and protocols. Available in both JSON and CSV formats.

STIX 2.1 Bundle

Export a STIX 2.1 threat intelligence bundle containing:

- **Infrastructure** objects for discovered OT devices
- **Indicator** objects for detected security findings
- **Relationship** objects linking indicators to infrastructure
- Standard STIX 2.1 envelope with proper UUIDs and timestamps

STIX bundles can be imported into threat intelligence platforms (MISP, OpenCTI, etc.) for correlation with other intelligence sources.

Topology Image

Export the current Cytoscape topology view as a PNG or SVG image via the save dialog.

Chapter 9: Session Management

Kusanagi Kajiki persists assessment data in a local SQLite database at `~/.kusanaginokajiki/data.db`. Sessions capture the full state of an assessment for later review or comparison.

Save Session

From the Capture tab, click **Save Session**. Provide a name and optional description. The session stores all assets, connections, protocol statistics, and analysis results. Sessions can be loaded later to restore the complete state.

Load Session

Select a previously saved session from the session list. Loading a session replaces the current state with the stored assets, connections, and topology. The current session name appears in the Capture tab.

.kkj Archive Export / Import

For portable sharing, export a session as a **.kkj archive** — a ZIP file containing a manifest.json and session.json with all assessment data. This format is self-contained and can be shared with colleagues or attached to assessment deliverables.

Import a .kkj archive to load a session from another analyst or system. The archive is extracted and loaded into the local SQLite database with a new session ID.

Delete Session

Remove a session from the local database. This action is permanent.

Chapter 10: Settings & Customization

Theme

Toggle between **Dark**, **Light**, and **System** (follows OS preference) themes using the toggle in the sidebar footer. The preference is persisted to `~/.kusanaginokajiki/settings.json`.

CLI Usage

Kusanagi Kajiki supports command-line arguments for integration with scripts and workflows. See Chapter 11 for the full reference.

Plugin Directory

The plugin system scans `~/.kusanaginokajiki/plugins/` for directories containing a `manifest.json`. Discovered plugins are listed in the Settings view. The plugin architecture supports stubs for signature packs, importers, exporters, and analyzers. Full plugin execution is reserved for a future release.

File Locations

File	Location	Purpose
Database	~/.kusanaginokajiki/data.db	SQLite session storage
Settings	~/.kusanaginokajiki/settings.json	User preferences (theme)
Plugins	~/.kusanaginokajiki/plugins/	Plugin manifest directory
Signatures	src-tauri/signatures/	YAML signature files
OUI Database	src-tauri/data/oui.tsv	IEEE MAC vendor lookup
GeoIP Database	src-tauri/data/dbip-country-lite.mmdb	IP geolocation

Chapter 11: CLI Reference

Kusanagi Kajiki accepts command-line arguments via clap 4. All arguments are optional — launching without arguments opens the GUI normally.

```
kusanaginokajiki # Launch GUI
kusanaginokajiki --open capture.pcap # Open PCAP on startup
kusanaginokajiki --open assessment.kkj # Open session archive
kusanaginokajiki --import-pcap file.pcap # Import PCAP on startup
```

Flag	Argument	Description
--open	<path>	Open a PCAP file or .kkj session archive on startup
--import-pcap	<path>	Import a PCAP file into the current session on startup
--help		Display help message with all available flags
--version		Display application version

CLI processing is deferred via `tauri::async_runtime::spawn` to allow the window to initialize before file processing begins. A 500ms delay ensures the frontend is ready to receive data.

Appendix A: Protocol Reference

Complete reference of all 19 protocols detected by Kusanagi Kajiki, with port numbers, detection method, and notes.

Protocol	Port(s)	Detection	Standard / Notes
Modbus TCP	502	Deep parse	MBAP + FC analysis, Device ID extraction
DNP3	20000	Deep parse	IEEE 1815, master/outstation, FC 130
EtherNet/IP	44818, 2222	Signature	CIP, Rockwell / Allen-Bradley
BACnet/IP	47808	Signature	ASHRAE, building automation
S7comm	102	Signature	Siemens S7 PLCs
OPC UA	4840	Port + Sig	OPC Foundation, binary/XML encoding
IEC 60870-5-104	2404	Port	Power grid SCADA telecontrol
PROFINET	34962-34964	Port	Siemens / PROFIBUS International
MQTT	1883, 8883	Port	IIoT gateways, 8883 = TLS
HART-IP	5094	Port	Process instrumentation
FF HSE	1089-1091	Port	Foundation Fieldbus HSE
GE SRTP	18245-18246	Port + Sig	GE Automation PLCs
Wonderware	5007	Port + Sig	AVEVA SuiteLink
HTTP/HTTPS	80, 443	Port	Web interfaces, HMI dashboards
DNS	53	Port	Name resolution
SSH	22	Port	Secure remote access
RDP	3389	Port	Remote desktop (Windows)
SNMP	161, 162	Port	Network management
Telnet	23	Port	Unencrypted remote access

Appendix B: ATT&CK; Technique Reference

The following MITRE ATT&CK; for ICS techniques are detected by Kusanagi Kajiki's analysis engine. For full technique descriptions, see attack.mitre.org/matrices/ics/.

T0855 — Unauthorized Command Message [Critical / High]

An adversary may send unauthorized command messages to instruct control system assets to perform actions outside their intended functionality. Kusanagi Kajiki detects this as Modbus write function codes (FC 5, 6, 15, 16) sent to broadcast unit IDs (0 or 255) or from a single source to 5 or more targets.

T0814 — Denial of Service [High]

An adversary may perform a denial of service to disrupt expected device functionality. Detected when Modbus FC 8 (Diagnostics/Restart) is observed from a non-engineering workstation, which could indicate an attempt to restart or disrupt PLCs.

T0846 — Remote System Discovery [High]

An adversary may attempt to get a listing of other systems on the network. Detected when an unknown or IT-classified device is observed polling 3 or more PLCs/RTUs, suggesting network reconnaissance activity.

T0856 — Modify Alarm Settings [Medium]

An adversary may modify alarm settings to prevent alerts from reaching operators. Detected when DNP3 unsolicited response messages (FC 130) are sent to a device not previously identified as a master station.

T0886 — Remote Services [Medium]

An adversary may use remote services to gain access to other systems. Detected when direct communication is observed between Purdue Level 1 (PLCs/RTUs) and Level 4 (Enterprise IT), bypassing the expected DMZ segmentation. L1↔L4 direct communication is rated Medium; L2↔L4 is rated Low.

Appendix C: Troubleshooting

"Permission denied" on live capture

Cause: Live capture requires elevated privileges to access network interfaces in promiscuous mode.

```
Run as root/administrator, or on Linux, grant capabilities:
sudo setcap cap_net_raw,cap_net_admin=eip target/release/kusanaginokajiki
```

"No interfaces found"

Cause: The libpcap library cannot enumerate network interfaces.

```
Verify libpcap is installed:
Fedora: sudo dnf install libpcap-devel
Ubuntu: sudo apt install libpcap-dev
macOS: brew install libpcap
Windows: Install Npcap with WinPcap compatibility
```

Build fails with missing system libraries

Cause: Tauri 2.0 requires WebKit2GTK and related libraries on Linux.

```
Install all required dependencies:
Fedora: sudo dnf install webkit2gtk4.1-devel libsoup3-devel
javascriptcoregtk4.1-devel
Ubuntu: sudo apt install libwebkit2gtk-4.1-dev libappindicator3-dev librsvg2-dev
```

Frontend not loading / blank window

Cause: The frontend build output must exist before the Rust backend can serve it.

```
Run npm run build before cargo check or npm run tauri dev.
The build/ directory must exist at the project root.
```

npm install fails with peer dependency errors

Cause: Svelte 5 and Vite have a peer dependency conflict on some systems.

```
Use: npm install --legacy-peer-deps
This is required on Fedora and some RHEL variants.
```

Database errors or corruption

Cause: The SQLite database at ~/.kusanaginokajiki/data.db may be corrupted or inaccessible.

```
Verify the directory exists and is writable:
ls -la ~/.kusanaginokajiki/
To reset: delete data.db (warning: destroys all saved sessions).
The database uses WAL mode and FK enforcement for reliability.
```

Wireshark not detected

Cause: Kusanagi Kajiki searches PATH and well-known installation directories.

```
Ensure Wireshark is installed and in your system PATH.
Common locations checked: /usr/bin/wireshark, /usr/local/bin/wireshark,
/Applications/Wireshark.app/Contents/MacOS/Wireshark (macOS)
```

Signatures not loading

Cause: The signature engine looks for YAML files in src-tauri/signatures/.

```
Verify the signatures directory exists and contains .yaml files.
Use the Signature Editor to reload signatures from disk.
```