IDIAP RESEARCH REPORT

TRACY CANVAS: A CRIMINAL NETWORK VISUALIZATION TOOL

Alejandra Sanchez Lara
Petr Motlicek
Dairazalia Sanchez-Cortes
Pradeep Rangappa
Srikanth Madikeri
Driss Khalil

Idiap-RR-03-2025

Version of JULY 28, 2025

Centre du Parc, Centre du Parc, Rue Marconi 19, CH - 1920 Martigny
T +41 27 721 77 11 F +41 27 721 77 12 info@idiap.ch www.idiap.ch

TRACY Canvas: A Criminal Network Visualization Tool

Alejandra Sanchez Lara[1,], Petr Motlicek[1,2] Dairazalia Sanchez-Cortes[1], Pradeep Rangappa[1], Srikanth Madikeri[3], and Driss Khalil[1]

[1]Idiap Research Institute, Martigny, Switzerland
[2]Brno University of Technology, Czech Republic
[3]University of Zurich, Switzerland
{alejandra.sanchezlara, petr.motlicek , dairazalia.sanchez-cortes,pradeep.rangappa }@idiap.ch
{srikanth.madikeriraghunathan}@uzh.ch

Abstract. Modern criminal investigations are increasingly challenged by the large volume and diversity of multimodal data, including audio, text, and metadata from intercepted communications. Building upon the Autocrime platform (a multimodal platform for combating organized crime) developed under the ROXANNE project, this report presents TRACY Canvas, a visualization tool developed entirely using open-source technologies to render complex communication networks. TRACY Canvas enables the exploration of diverse network types, including teletelephone networks, speaker networks from Autocrime outputs, terminal-based networks, and individual level networks with concurrent physical presence. These capabilities support behavioral and topological analysis of suspects within criminal networks. The tool offers features such as temporal filtering, dynamic edge rendering, and metadata overlays, allowing law enforcement agencies to interactively analyze both content and non-content signals. Validated on real and simulated cases, our tool connects backend analytics (i.e., Autocrime platform) with an intuitive front-end (i.e., visualisation tool) to support efficient and transparent criminal investigations.

Keywords: TRACY · ROXANNE · Law Enforcement Agencies · Suspect Detection· Mobile Signaling Data · Autocrime

1 Introduction

Organized crime remains one of the most difficult and resource-intensive forms of criminal activity to investigate, posing significant risks to public safety and national security. Law enforcement agencies (LEAs) are increasingly challenged by the scale, sophistication, and transnational nature of such networks, which often leverage modern communication technologies to evade detection. To support the growing complexity of criminal investigations, we propose **TRACY Canvas**, a new open-source visualization tool developed by Idiap Research Institute (Martigny, Switzerland) to render and analyze enriched communication networks.

TRACY Canvas was initially created to replicate and extend the functionality of the Frontend Visualization Tool (FVT) developed under the ROXANNE project [1] an EU Horizon 2020 initiative focused on content-based analysis of intercepted communications using technologies such as speaker diarization, speech recognition, and named entity extraction [Fabien et al., 2021, Madikeri et al., 2025].

FVT provided visual exploration capabilities but was proprietary, closed-source, and built on a rigid JavaScript framework that restricted its usability and modifiability [ROXANNE Consortium]. In response, Idiap initiated the development of a visualization tool using standard open source Python libraries. The goal was to ensure full access to the visualization layer, enabling internal teams to integrate new features, adapt the tool to specific investigative needs, and debug or extend it independently of third-party constraints.

The tool was later expanded under the TRACY project (A bigdata analyTics from base-stations Registrations And CDRs e-evidence sYstem)[2] to support new types of evidence from *non-content data (NCD)*—metadata such as subscriber details, traffic/logs, and location information—alongside supporting evidence like surveillance footage and witness testimonies. These include behavioral scoring, shared device usage, and detecting when individuals were physically present in the same location. This aligns with recent work in TRACY focused on large-scale processing of synthetic but realistic data to track suspect movements and improve offender identification using diverse NCD sources [Rangappa et al., 2024]. Such data is especially useful when speech or text content is missing, encrypted, or not enough to infer behavior.

This report presents the design and implementation of **TRACY Canvas**, highlighting how it integrates both content and non-content data into a unified, interactive visualization tool. We describe its architecture, core features such as temporal filtering, behavioral overlays, and metadata inspection and demonstrate its application across realistic investigative case studies (Sections 4.1–4.2).

2 Motivation

The development of TRACY Canvas was driven by the need to create an open, extensible, and fully integrated visualization tool that could address the growing complexity of criminal investigations and overcome FVT visualization created under the ROXANNE project that offered visual exploration capabilities but was proprietary and closed-source. Built on a JavaScript framework, it limited usability, adaptability, and integration making it impossible to extend, customize, or align with evolving research and operational needs.

At the same time, the TRACY project introduced new analytical models, especially focused on non-content data (NCD) such as behavioral scoring, co-usage of devices, and concurrent physical presence [Rangappa et al., 2024]. However, there was no existing tool that could support visualizing or interpreting these

[1] `https://www.roxanne-euproject.org/`
[2] `https://www.tracy-project.eu/`

outputs. Moreover, maintaining backward compatibility with ROXANNE outputs was essential to ensure continuity across EU-funded research efforts.

To address these gaps, TRACY Canvas was built from the ground up using standard Python libraries. This design choice ensured full ownership of the codebase, flexibility to incorporate future features, and seamless integration into Python-based investigative workflows. The tool enables visual exploration of complex communication networks by unifying content (e.g., transcripts, speaker IDs) and non-content evidence (e.g., physical co-location), filling a critical gap left by FVT.

To meet the specific challenges faced in real-world investigations, TRACY Canvas was designed with several key features that directly address operational and analytical needs:

- *Temporal Filtering*: Allows analysts to isolate and focus on communication events within specific 15-minute windows, making it easier to track bursts of activity or detect co-location patterns.
- *Dynamic Edge Rendering*: Encodes metadata such as interaction frequency or behavioral risk through variations in edge width or color, helping to visually highlight important relationships in complex networks.
- *Behavioral Score Overlays*: Incorporates outputs from Autocrime by overlaying behavioral risk scores onto the network, enabling investigators to quickly prioritize suspicious interactions.
- *Node-Level Metadata Inspection*: Provides on-demand access to detailed node information (e.g., speaker ID, gender, social influence score, etc) via interactive click and hover actions, maintaining context without disrupting the analytical flow.

These features demonstrate how TRACY Canvas bridges the gap between data complexity and investigator usability, as illustrated in realistic case studies such as the ROXSD Scenario and the Robbery case (Sections 4.1–4.2), where both content and non-content data are effectively visualized and analyzed.

3 Methodology

The development of TRACY Canvas followed an iterative and user-centered methodology, with the goal of creating an interactive visualization tool that could replicate and extend the ROXANNE GUI while being fully open, maintainable, and built entirely in Python. One of the primary design constraints was the exclusion of JavaScript libraries in order to maximize accessibility, portability, and ease of integration within standard investigative environments. This decision ruled out many high-end front-end libraries, necessitating a careful evaluation of Python-native visualization frameworks. The development proceeded in three phases described in Figure 1.

3.1 Input, Output, and Network Generalization

TRACY Canvas operates on structured JSON files originally defined within the ROXANNE project as outputs of the Autocrime backend. These files represent

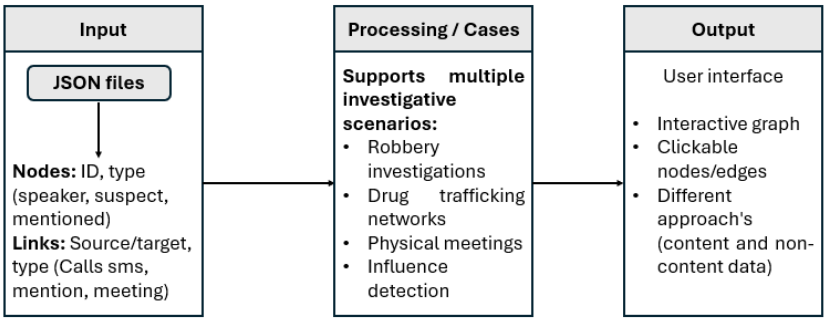

Fig. 1: System architecture of TRACY Canvas

multimodal communication networks and serve as the input format for TRACY's interactive visualizations. A typical JSON file includes the following components:

- **nodes**: An array of entities such as speakers, phone numbers, suspects, or mentioned individuals. Each node should include at least an `id` and a `type` (e.g., `speaker`, `suspect`, `mentioned`), and may include additional metadata such as `gender`, `phone_number` when available (otherise anonymized terminal identifier), or cluster assignments for community detection.
- **links**: A list of interactions between nodes. Each link should specify:
  - Source and target identifiers: `id_from` / `id_to` (for speaker networks) or `phone_from` / `phone_to` (for terminal or telephone networks),
  - `type` of interaction: `phone_call`, `sms`, `mention`, `concurrent_meeting_edge`,
  - `date` timestamp,
  - Optional metadata: `transcript`, `topic`, `language`, `call_id`, and geolocation fields for meetings.

The visualization tool builds directly on this existing structure, enabling dynamic, multimodal, and interactive visualizations without modifying the underlying schema. Certain data handling steps such as slot-based timestamp segmentation, pair detection, and visualization oriented graph construction extend the use of the original outputs for advanced visual inspection.

The output of TRACY Canvas is an interactive interface which allows users to visualize and explore the network dynamically. This includes node and edge highlighting, time-based filtering, edge thickness by frequency, behavioral scores, NER overlays, and access to related audio and transcript metadata.

While originally designed for criminal investigations, the interface is generalizable to any graph-based dataset conforming to the same JSON schema. Potential use cases include corporate communication analysis, fraud detection, epidemiological contact tracing, or general social network analytics. For instance, FraudVis demonstrates how unsupervised fraud detection outputs can be effectively interpreted through coordinated, multi-view visual interfaces that support temporal analysis, inter-group comparison, and individual user inspection [Sun et al., 2018]

3.2 Design Constraints and Development Process

Developing TRACY canvas exclusively in Python presented several technical and design challenges. One of the main constraints was the requirement to replicate the functionality of the Autocrime visualization tool developed for the ROXANNE project, while intentionally avoiding JavaScript. This decision was driven by the goal of creating an open-source tool that could be easily modified, extended, and maintained within Python-based research environments. While this approach improved accessibility for data scientists and researchers familiar with Python, it also meant working without the wide range of specialized JavaScript front-end libraries typically used for complex, interactive network visualizations, requiring custom development to achieve similar functionality.

During development, the visualization tool was tested across three different Python-based libraries and frameworks, including:

- **Streamlit-Agraph:** is a powerful and lightweight library for visualizing networks/graphs. Provided a quick prototype with basic node interaction using the `streamlit-Agraph` component [Klose, 2020], but lacked layout control and flexibility for investigative use cases.
- **Pyvis:** is a Python module that enables visualizing and interactively manipulating network graphs in the Jupyter notebook, or as a standalone web application. Pyvis is built on top of the powerful and mature VisJS JavaScript library, which allows for fast and responsive interactions while also abstracting away the low-level JavaScript and HTML [Perrone et al., 2020]. While visually appealing and easy to use, it showed instability with large graphs and limited advanced interaction capabilities.
- **Dash-Cytoscape + HTML:** a Python library for interactive network visualization built on top of Cytoscape.js and integrated into the Dash web framework [Plotly Technologies Inc., 2024]. It provides extensive customization and interactivity options entirely within Python, making it suitable for research workflows and rapid prototyping. This solution was ultimately chosen for the final version of the tool due to its strong support for dynamic graph interaction, seamless compatibility with network data, and full control within the Python ecosystem—key requirements for replicating the investigative capabilities of the ROXANNE Autocrime platform [Madikeri et al., 2025, Fabien et al., 2021].

Another key challenge was ensuring visual layout consistency especially when filtering nodes or switching between graph types. This was addressed by implementing state preservation and layout memory across views.

3.3 Initial Prototype: Streamlit with Agraph

The initial prototype of the TRACY Canvas was developed using **Streamlit** as the interface framework and **Agraph** for rendering network graphs. This setup allowed for rapid development and offered a basic level of interactivity with nodes

and edges. It served as a proof of concept for replicating ROXANNE's graphical outputs using only Python-based technologies. However, several limitations quickly became apparent:

- **Limited Event Handling:** Agraph did not support multiple simultaneous event listeners. For example, clicking on an edge could trigger only a single action, making it difficult to enrich the interaction with features such as displaying metadata, highlighting connected nodes, or loading call transcripts dynamically.
- **Inconsistent Network Layouts and NER Misrepresentation:** When visualizing the provided JSON files, the resulting networks did not resemble the structured and semantically organized views presented in ROXANNE project. Node positions were rendered arbitrarily, leading to a disorganized and difficult-to-interpret graph structure. Additionally, when attempting to display Named Entity Recognition (NER) information using a SpaCy model within Streamlit, the system incorrectly re-analyzed the text instead of respecting the already annotated NER entities included in the JSON files. This led to discrepancies in recognized entities and misalignment with the metadata used by the ROXANNE project, making the reproduction of accurate visual and analytical outputs infeasible.
- **Limited Customization Without JavaScript:** While Agraph offers some flexibility, advanced features such as changing node shapes, adjusting edge styles based on metadata or introducing custom layouts required JavaScript-level intervention. Since one of the project constraints was to work entirely in Python, this limited the scalability and extensibility of the tool.

Figure 2 and Figure 3 illustrated the initial prototype created.

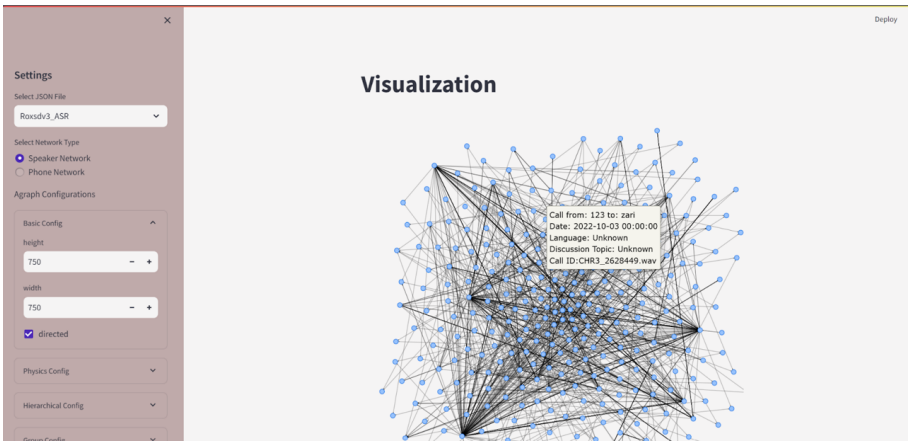

Fig. 2: Prototype of network view using Streamlit and Agraph. This prototype facilitated rapid development and testing but lacked advanced layout control and interaction features needed for investigative analysis.

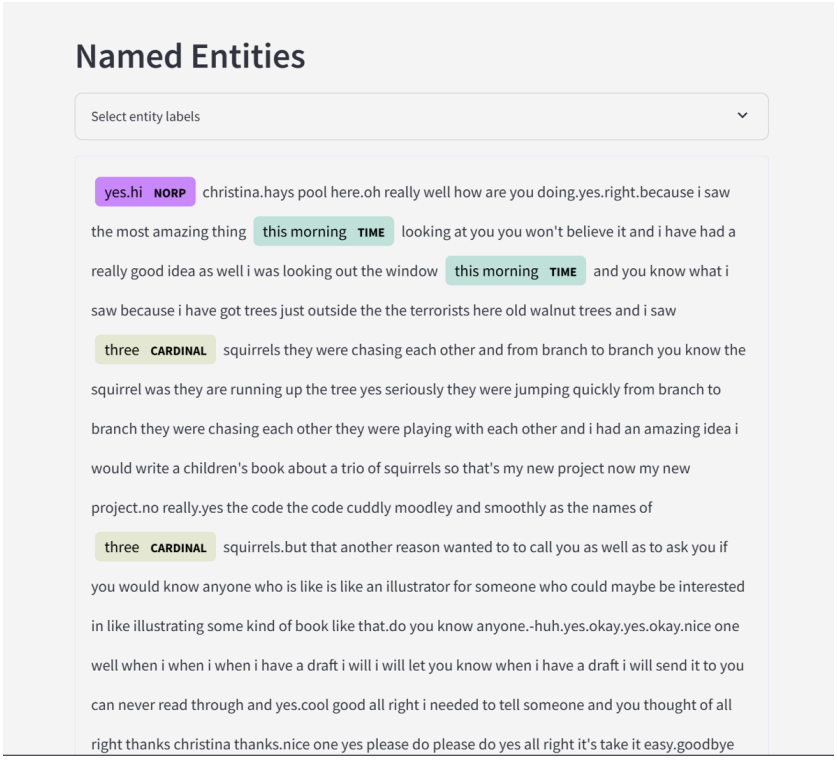

Fig. 3: NER functionality in action: The interface highlights different named entities (e.g., NORP, TIME, CARDINAL) within a block of text, indicating the model's ability to extract structured information.

3.4 Second Prototype: Pyvis and Streamlit

Following the limitations encountered with Agraph, a second prototype was developed using **Pyvis** for graph rendering in combination with **Streamlit** for the user interface. The goal was to enhance the visual quality and interactivity of the networks while remaining within a fully Python-based environment.

This version introduced cleaner visuals and allowed for some level of node interaction. For example, it supported basic speaker and telephone network generation and enabled users to select nodes and retrieve call information. However, several limitations persisted:

– **Limited Edge Interaction:** Pyvis lacks native support for interactive edge features such as tooltips or dynamic metadata display without custom JavaScript. As a result, it was not possible to replicate key functionalities of the original FVT visualization tool such as displaying timestamps, call topics, or transcripts when interacting with edges in the network.

- **Click Events Were Node-Only:** Interactivity was restricted to nodes. Clicking an edge had no effect unless JavaScript was integrated. Even node-based click actions could only display simple metadata and could not trigger layout or style changes.
- **Styling Constraints:** While some node coloring was possible, dynamically changing colors or shapes based on user interaction (e.g., highlighting a node or its neighbors) required JavaScript, which went beyond the project's Python-only constraint.
- **Static Layouts and Limited Control:** Layout customization was minimal, and large networks (such as full telephone networks) became unreadable due to overlapping nodes and lack of responsive force-directed layout control.

Despite these constraints, this iteration helped validate several visual assumptions, such as node sizing, hover text, and the representation of speaker-phone relationships. Figure 4, Figure 5 and Figure 6 illustrated the second prototype created.

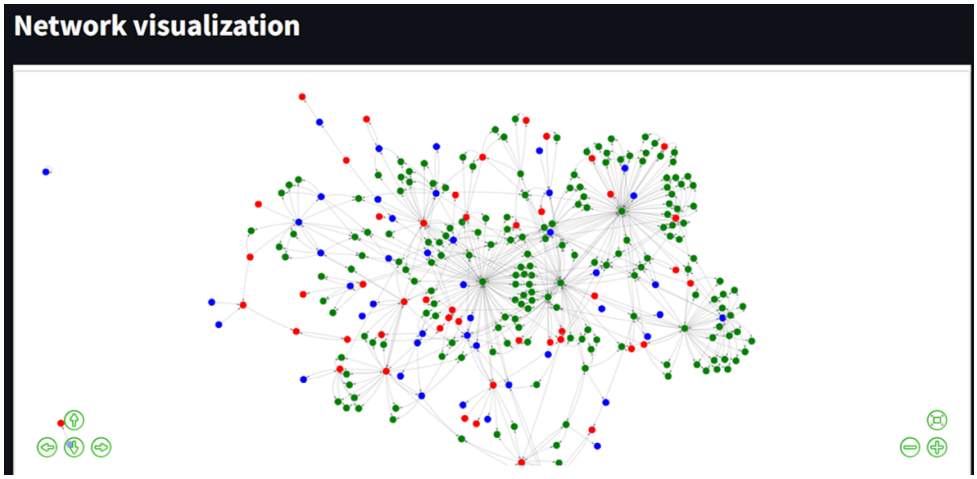

Fig. 4: Visualization of the speaker network using Pyvis and Streamlit. Nodes are color-coded by type (e.g., suspects, speakers, mentioned entities), and the layout reveals community clusters and connectivity patterns. This prototype introduced improved grouping and visual separation compared to earlier versions.

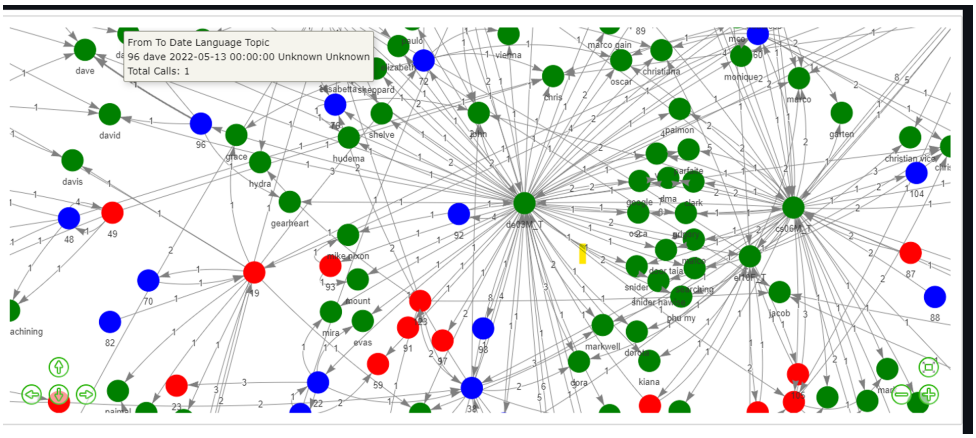

Fig. 5: Extending functionality for speaker-phone relationship analysis. Node labels display speaker identifiers, while edge thickness reflects the number of interactions. Hovering over an edge reveals call metadata such as timestamp, topic, and file ID, enabling deeper inspection of communication patterns.

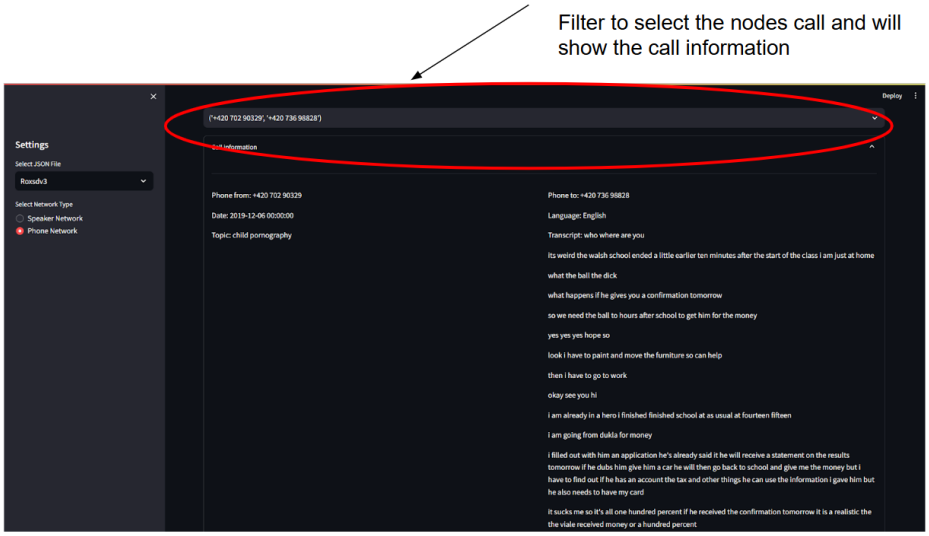

Fig. 6: Third prototype displaying enhanced call metadata, including phone numbers, topics, timestamps, and transcripts. While this version introduced a filter for exploring interactions between specific phone numbers, it lacked practical edge selection—edges were inferred indirectly from node connections rather than explicitly clickable—limiting usability. The absence of native edge interactivity and need for JavaScript-based logic remained a constraint.

3.5 Third Prototype: Dash-Cytoscape (v1.0)

The final version of the visualization tool was developed using **Dash-Cytoscape**, offering a complete, scalable, and interactive solution entirely in Python. This implementation overcame the core limitations of previous prototypes (Streamlit, Agraph and Pyvis), enabling fine-grained control over layout, event handling, and visual customization, while still avoiding the need for JavaScript.

This version was designed to mirror the full range of interactive capabilities previously seen in the visualization tool developed for ROXANNE project, with additional flexibility, maintainability, and extensibility.

Multiple Network Views The visualization tool supports four distinct types of network representations:

- **telephone network:** Visualizes phone-terminal connections related to suspects, including interactions with other individuals, along with details such as caller/callee IDs, timestamps, call language, topic, and social influence scores.
- **Speaker Network:** Displays connections between speakers, including both direct participants and names mentioned in audio. Includes outlier identification and influence metrics.
- **Phone-Speaker Network:** Combines phone and speaker data for a detailed mixed network view.
- **Speaker-telephone network:** Focuses on speakers and shows the phone numbers they use.

Key Features

- **Filter and Network Selection**Users can dynamically switch between the four network types. A filtering tool enables analysts to isolate and focus on specific individuals or links.
- **Click Interactions (Node and Edge Events):** Clicking a node reveals its ID, gender, associated speaker, outlier status, and social influence score. Clicking an edge displays detailed call metadata including caller/callee, datetime, language, topic, audio ID, transcript with Named Entity Recognition (NER), and an embedded audio playback component.
- **Mouse-over Events:** Hovering over edges triggers display of quick metadata such as participants, language, and topic — enabling rapid inspection without clicks.
- **Community Detection and Visual Encoding:** In the Speaker Network, node colors are dynamically updated based on detected communities, allowing users to visually identify clusters and communication subgroups.
- **Consistent Interaction Logic:** All interaction behaviors are consistently applied across all four network types, ensuring a smooth user experience.

Figure 7 and Figure 8 illustrated the third protoype created.

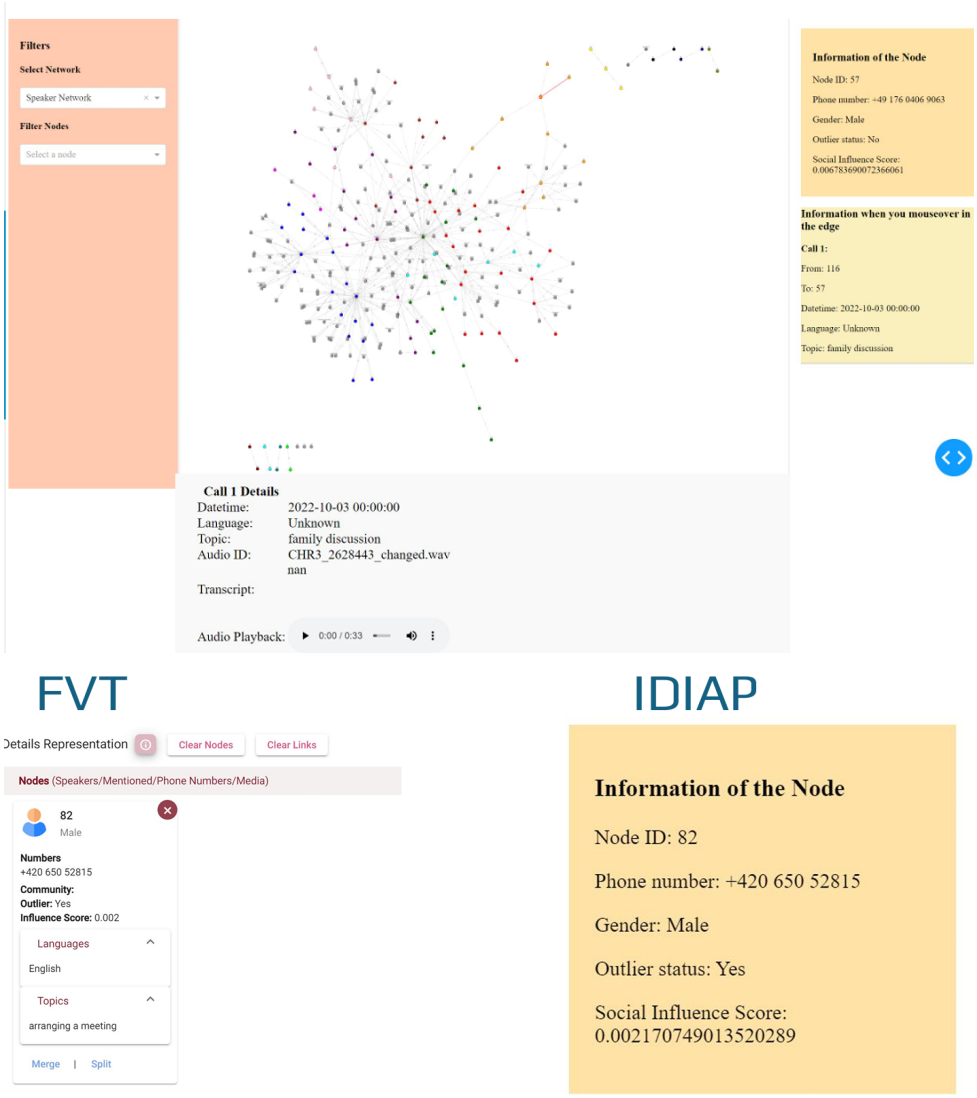

Fig. 7: Final TRACY canvas interface and a comparison with the original FVT tool. The top section illustrates the interactive network with filters, node/edge metadata, and embedded audio playback. The bottom panels compare how node information is displayed in FVT visualization tool versus the updated version developed by IDIAP, highlighting improved clarity, interactivity, and integration of analytical features.

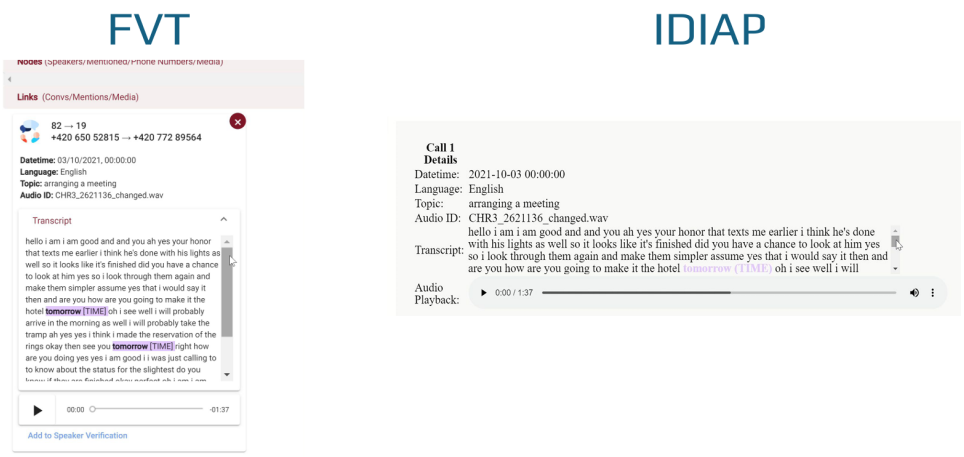

Fig. 8: Example of node and edge metadata shown during click events, including social influence score and call details.

3.6 Final: Dash-Cytoscape (v2.0)

While the core structure of the user interface—consisting of three main panels (filters, network visualization, and metadata) was already established in Version 1.0, the release of Version 2.0 introduced a major visual redesign. The updated interface focused on enhancing graphical consistency, color schemes, and overall aesthetic appeal.

Key improvements included:

- A cleaner and more cohesive visual style with a modernized color palette, enhanced contrast, and improved readability.
- Redesigned buttons and widget spacing for a more professional and intuitive interface.
- Better visual separation and alignment across interface sections, enhancing navigation and reducing cognitive load.
- The introduction of a filter to remove "mentioned" nodes (represented as triangles), helping users focus on directly involved participants.

The last version of TRACY Canvas (v2.0) can be appreciate it in Figure 9.

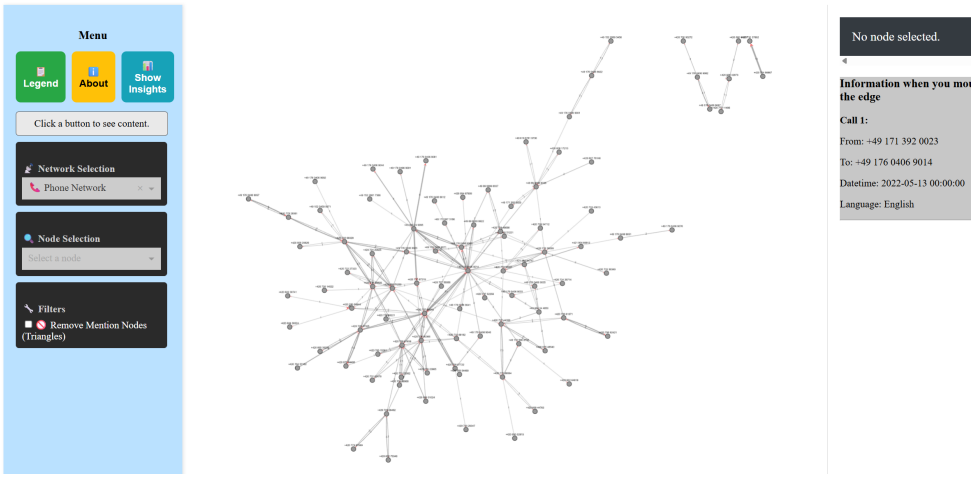

Fig. 9: User interface introduced in Version 2.0, featuring a visually refined layout with enhanced spacing, modernized color palette, and improved usability. While the three-panel structure remained consistent with earlier versions, this release added new filtering capabilities—such as the option to remove "mentioned" nodes (triangular icons)—and introduced clearer navigation and visual alignment across the interface.

4 Visualization

TRACY Canvas introduces several critical enhancements. Most notably, it supports new edge types that capture concurrent physical presence events, enriching the analysis of co-location patterns. Additional improvements include a time range filtering interface, automatic identification of suspect nodes, clearer differentiation between communication edge types, and an integrated insights panel for high-level summary statistics. These features significantly enhance the usability and analytical depth of the tool.

Libraries Used for Final Visualization: The development of this tool relied on a combination of Python libraries, each playing a specific role in data processing, analysis, and visualization:

- **Dash, Dash Cytoscape, Dash Leaflet, Dash Table** — Dash is a Python library for building interactive web applications. Dash Cytoscape enables complex network visualizations; Dash Leaflet adds geospatial mapping capabilities; and Dash Table provides sortable and filterable data tables. This stack, maintained by [Plotly Technologies Inc., 2024], was chosen for its flexibility and ability to add rich interactive features without relying on JavaScript libraries.
- **pandas** — A powerful data analysis and manipulation library, used to structure and filter node and edge data, convert it to graph formats, and generate tabular summaries [McKinney, 2010].

- **spaCy** — An industrial-strength natural language processing (NLP) library, used for applying Named Entity Recognition (NER) to call transcripts, allowing automatic extraction of entities like names, locations, and dates [spaCy, 2020].
- **Built-in Python modules:** `json`, `collections` (e.g., `defaultdict`, `Counter`), `datetime`, `timedelta`, `random`, `copy`, `re`, `os`, and `flask` were used for JSON parsing, time filtering, regular expression processing, file operations, and basic backend logic.

4.1 ROXSD Scenario

One of the key evaluation scenarios used to validate the visualization tool derives from the **ROXSD dataset**, a simulated but realistic multimodal dataset developed under the ROXANNE project [Motlíček et al., 2024]. ROXSD models the communications of a fictional criminal network engaged in drug trafficking, monitored by multiple (fictional) law enforcement agencies through phone taps, multimedia seizure, and dark web surveillance.

In this setting, 13 speakers were *enrolled*, meaning their identities were known and their voice profiles were available for speaker identification and scoring modules. This subset was extracted from a larger pool of 95 speakers. To simulate realistic investigative challenges, only two enrolled speakers were assigned two phone numbers each, while all others received one. Phone numbers for unknown speakers were either selected from unassigned slots or generated randomly, with country codes preserved to maintain contextual diversity. In multi-speaker calls, a single phone number was assigned per side to simplify the network structure.

This setup captures the ambiguity and complexity typical of real-world investigations. TRACY Canvas enabled exploration of this scenario by displaying speaker-phone relationships, identifying community clusters, and linking behavioral scores to specific interactions. Investigators could analyze how enrolled and unknown speakers connect, examine communication patterns, and test hypotheses regarding speaker influence or network position. We visualize four main networks in this case:

telephone network: Captures all phone-based connections between suspects. Each edge includes metadata such as caller and callee IDs, timestamp, language, topic, and social influence scores. Figure 10 illustrates the node-level metadata available on selection.

Speaker Network: Represents connections between speakers, including both direct interactions and references made in intercepted audio. A special edge type ("mentioned") is shown in blue to indicate when a speaker is mentioned in a conversation without being an active participant. Figure 11 demonstrates these distinctions.

Phone–Speaker Network: Provides a hybrid view linking phone numbers to speaker identities. This view helps identify which speaker is using a specific phone number during a communication event (see Figure 12).

Speaker–telephone network: Reverses the perspective to focus on speakers and the list of phone numbers associated with each one. This view is particu-

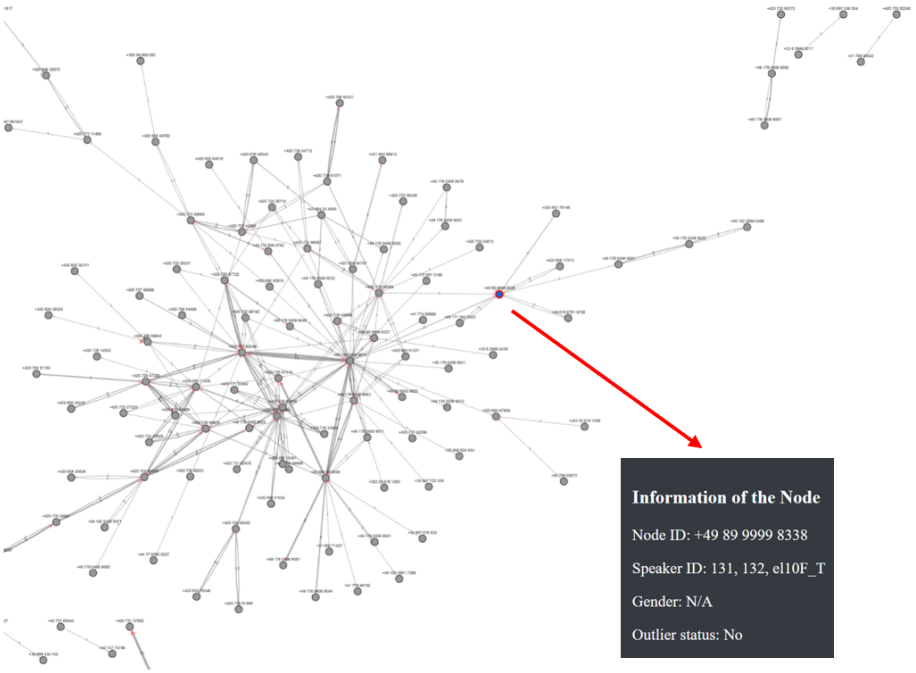

Fig. 10: Node Information: Displays metadata such as ID, gender, speaker ID, outlier status, and social influence score when a node is selected.

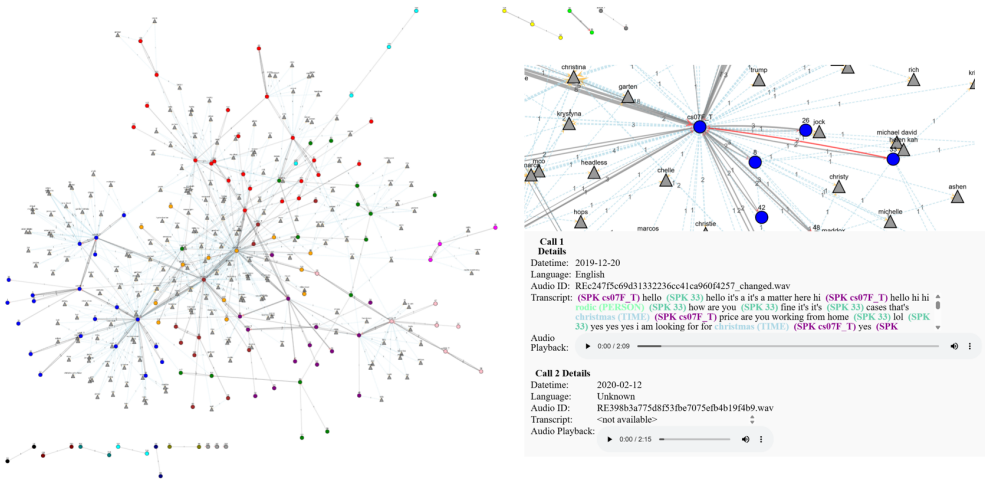

Fig. 11: Speaker Network with direct and mentioned connections. Includes audio metadata and transcripts.

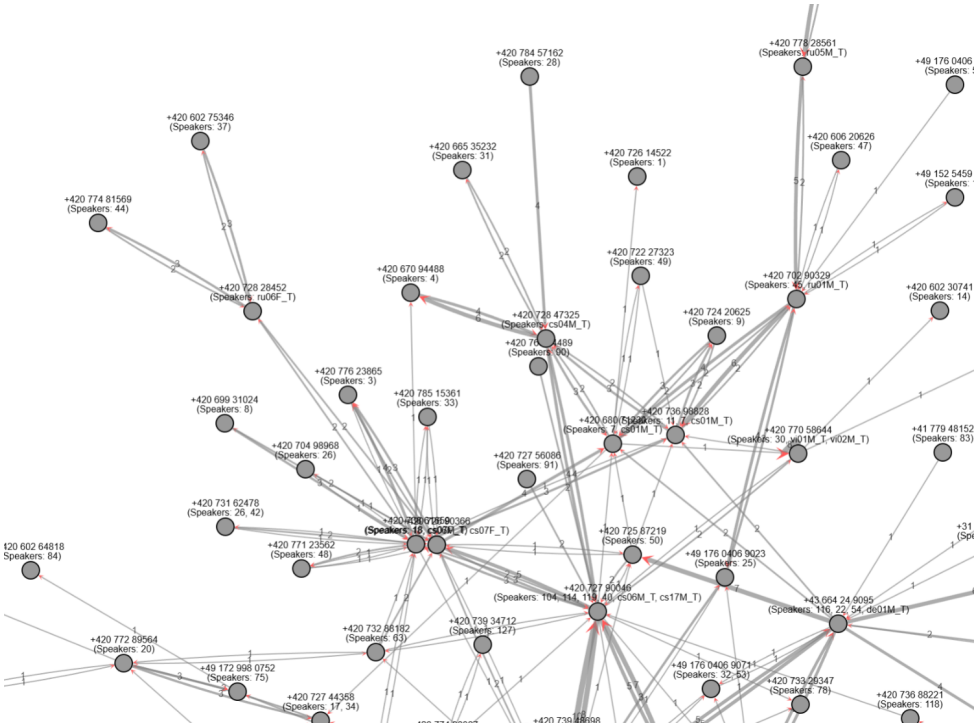

Fig. 12: Phone-Speaker Network: Indicates which speaker is using each phone number.

larly useful for identifying SIM card sharing, device switching, or alias behaviors (Figure 13).

Additionally, the tool includes interactive filters. For example, selecting the **Remove Mention Nodes** option hides triangular nodes representing mentioned-only speakers to reduce visual clutter.

The visualization tool ensures compatibility with ROXANNE input files, as it uses the same JSON structure, including NER results, audio events, and transcripts. This maintains continuity with previous backend data and results. Being based on Dash, the visualization system supports easier maintenance and customization over time, allowing adjustments without deep front-end (JavaScript) experience.

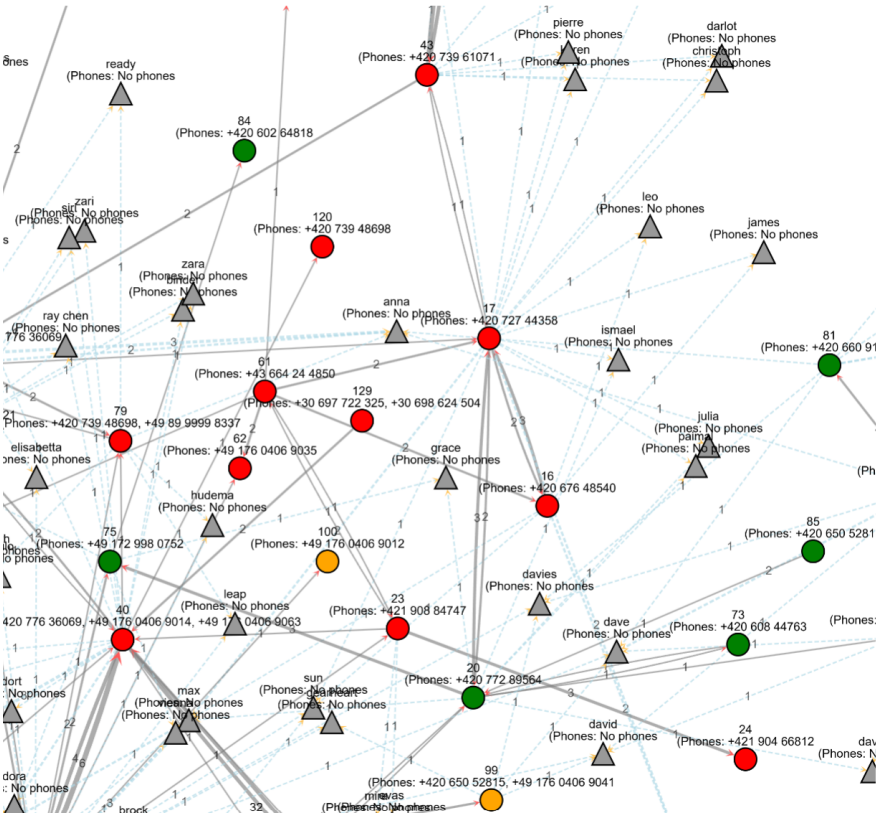

Fig. 13: Speaker-telephone network: Displays speakers and their associated phone numbers.

4.2 Robbery case

The demonstration case is based on legally extracted telecommunication registry data provided by the Hellenic Police (HPOL). This dataset includes various communication event types and anonymized spatial data designed to support real-world investigative scenarios.

Data Source. The dataset used in this case study was legally obtained from telecommunication service providers by the Hellenic Police (HPOL). It consists of registry-level metadata describing mobile device activity during the investigation period. Specifically, it includes three types of communication events:

- **CALL** — voice communication events.
- **SMS** — text messaging events.
- **DATA** — mobile data session records.

Data Composition. The dataset is heavily imbalanced in terms of event types, with the majority of records related to mobile data usage. Its structure is as follows:

- Approximately 95% of events correspond to **DATA** sessions.
- Around 5% are **CALL** events, and another 5% are **SMS** events.
- The dataset spans roughly 30,000 unique, anonymized terminal identifiers.
- **Note:** No content information (e.g., transcripts or audio) is available for CALL or SMS interactions.

Data Fields. The dataset includes several metadata attributes associated with each communication event. Key fields are:

- `cellid` — Identifier of the cellular tower involved in the event.
- `latitude`, `longitude` — Geographical coordinates of the corresponding cell tower.
- `measured_at` — Timestamp indicating when the interaction occurred.
- `numberA`, `numberB` (if available) — Anonymized identifiers of the participants involved in the communication.

Temporal and Spatial Resolution

- **Time frame:** July 28, 2023 from 08:00 to 12:00.
- **Spatial resolution:** Aggregated into 2 km × 2 km grid cells.
- **Privacy:** All data is anonymized and spatially aggregated to comply with legal and ethical standards.

This new implementation integrates the **Autocrime** analytical approach with the **TRACY** visualization tool to enhance the detection and interpretation of suspect behavior. Autocrime contributes a behavioral scoring mechanism that analyzes communication behavior by establishing links between individuals and

computing a *call score* a metric that quantifies the relevance or suspiciousness of each person based on their communication patterns.

By merging this scoring system with TRACY's spatial-temporal filtering capabilities, the visualization tool enables investigators to move beyond structural network analysis and toward behavioral prioritization. Analysts can now not only visualize the connections within the network but also highlight individuals whose interactions require deeper scrutiny due to their frequency, timing, or contextual anomalies.

Figures 15 and 16 showcase the latest implementation of this integration, displaying different edge types (calls, SMS, and concurrent physical presence) along with behavioral score overlays. This enriched visualization empowers law enforcement to perform targeted and context-aware investigations using both content and non-content signals.

This case highlights TRACY's capacity to process complex real-world datasets and to integrate multiple modalities—communication, geolocation, and metadata into a unified and interactive investigative interface. The following features were specifically integrated into the TRACY tool to support this demonstration:

- **SMS Edge**
  - A newly integrated feature allowing visualization of **SMS** separating edges that indicate calls and concurrent physical presence edges. This adds a orange edge type to the speaker network to identify the type of communication done between suspects. This relationship is shown in Figure 15.
- **Concurrent physical presence Edge**
  - A newly integrated feature allowing visualization of **concurrent physical presence** between suspects using cell ID data (location approximations). This adds a green edge type to the speaker network to map non-content-based interactions (physical proximity events). This is shown in Figure 15 and Figure 14.
- **Time filtering**
  - An additional feature integrated into the TRACY visualization tool is a time-based filter, designed to enhance the temporal analysis of suspect interactions. This filter allows analysts to isolate and examine communication events that occurred within specific 15-minute intervals. By selecting a start and end time from a list of predefined 15-minute blocks.
- **Edge width**
  - To enhance the interpretability of communication intensity within the network, the TRACY visualization tool incorporates dynamic edge width encoding. The width of each edge is proportional to the number of interactions between the connected suspects. This is shown in Figure 15

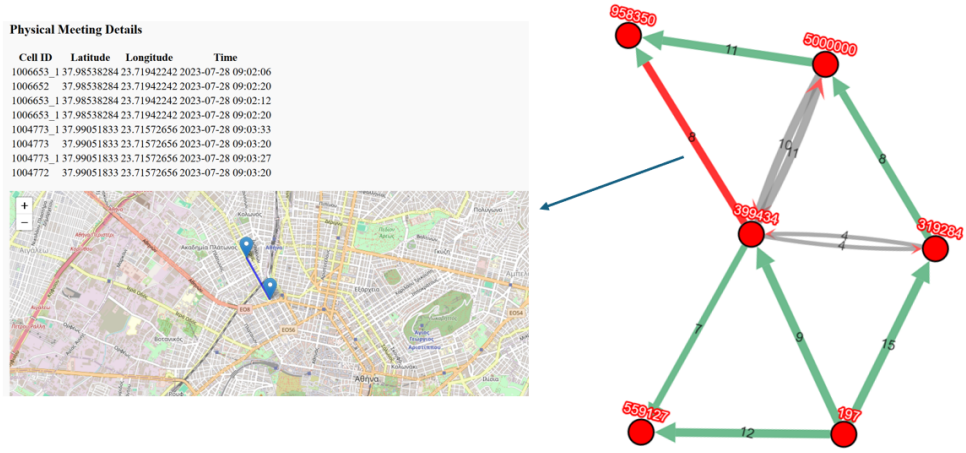

Fig. 14: Speaker-telephone network provides the speaker and the telephone number the speakers are using.

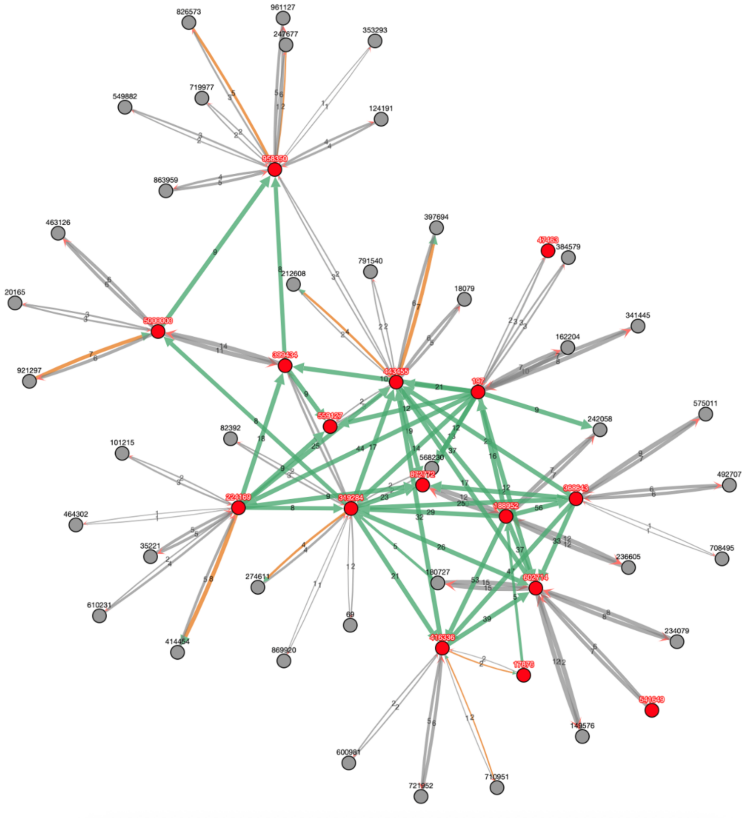

Fig. 15: HPOL data demonstrates the possibility of visualizing three different edge types: calls, SMS, and concurrent physical presence.

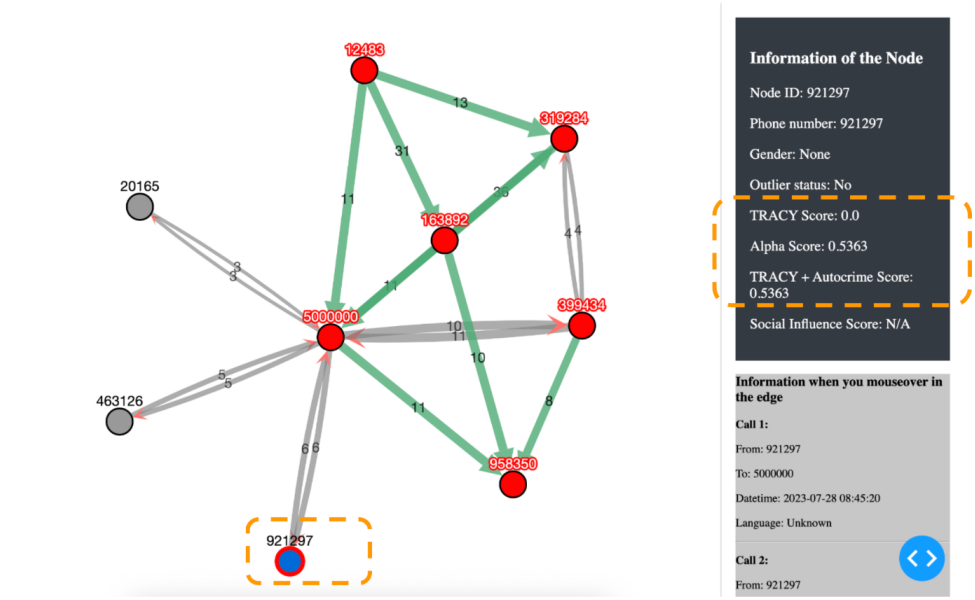

Fig. 16: Visualization of suspect identification scores.

5 Availability of TRACY Canvas

TRACY Canvas is available for security practitioners from EU as an open-source SW. More details can be found here.[3] It contains all necessary dependencies and configuration files, as well as a complete demonstration environment based on the TRACY's use-cases (see more at https://www.tracy-project.eu).This demo allows users to explore the full range of features, including network views, metadata panels, temporal filters, and behavioral scoring overlays.

6 Conclusions

This report presents TRACY Canvas, a visualization tool that serves as an open, modular, and extensible alternative to the proprietary tools previously developed under the ROXANNE project. The tool successfully bridges the gap between back-end multimodal analytics and front-end investigative usability by offering full ownership, reproducibility, and adaptability in both law enforcement and research settings.

Developed entirely in Python, Idiap replicates and extends the core functionalities of FVT while introducing critical new capabilities: multi-network

[3] `https://www.roxanne-euproject.org/tracy`

views, time-based filtering, dynamic edge weighting, and integration of behavioral scoring mechanisms from the Autocrime backend. The decision to eliminate JavaScript libraries not only aligns with operational constraints common in many public-sector contexts, but also enhances maintainability and simplifies deployment.

The visualization tool has been validated through multiple realistic case scenarios:

- The **ROXANNE case** demonstrated support for speaker-telephone mappings and behavioral prioritization using enrolled speaker metadata.
- The **Demo Robbery scenario** integrated anonymized telecom registry data with Autocrime scores, enabling suspect prioritization based on frequency and context of interactions.

TRACY Canvas empowers investigators to move beyond static, topology-based views and toward dynamic, behaviorally informed network analysis. By visualizing both content and non-content data streams, the visualization offers a rich foundation for investigative reasoning and hypothesis testing. Its successful deployment illustrates the feasibility of building high-impact analytical tools entirely within the Python ecosystem, ensuring transparency and long-term sustainability.

7 Future Work

While the current implementation of the visualization tool provides a robust and feature-rich environment for network exploration, several directions remain open for further development and enhancement.

- **User Authentication and Access Control:** Integrating user roles and permissions could allow for secure, multi-user environments where different stakeholders (e.g., analysts, supervisors, external reviewers) have access to different data views or functionalities.
- **Real-Time Data Integration:** The current system operates on preprocessed JSON input. Expanding the platform to connect with live data streams from interception systems, mobile operators, or surveillance tools would unlock real-time situational awareness capabilities.
- **User Collaboration and Audit Trails:** Future versions could support multi-user environments, where investigators can annotate nodes, share filtered views, or log investigative decisions. This would support transparency, repeatability, and collaborative workflows in intelligence or judicial contexts.
- **Scalability and Performance Optimization:** As datasets grow in size and complexity, optimizing rendering performance, caching, and server-side filtering will be essential for maintaining responsiveness.
- **Extension Beyond Criminal Intelligence:** Finally, the architecture is adaptable to domains beyond forensic analysis—including epidemiology, cybersecurity, and corporate network auditing. Future case studies will explore its applicability to these emerging use cases.

These developments will continue to position TRACY as a modular, transparent, and investigator-oriented tool for multimodal network analysis across diverse operational environments, such as:

8 References

References

Maël Fabien, Shantipriya Parida, Petr Motlicek, Dawei Zhu, Aravind Krishnan, and Hoang H. Nguyen. Roxanne research platform: Automate criminal investigations. In *Interspeech Show and Tell*, 2021.

Chris Klose. Showing off the streamlit-agraph component. `https://discuss.streamlit.io/t/showing-off-the-streamlit-agraph-component/6712`, 2020. Accessed: 2025-06-02.

Srikanth Madikeri, Petr Motlíček, Jakub Tkaczuk, Pradeep Rangappa, Alejandra Sanchez Lara, Johan Rohdin, Dawei Zhu, Aravind Krishnan, Dietrich Klakow, Zahra Ahmadi, Marek Kováč, Dominik Boboš, Costas Kalogiros, Andreas Alexopoulos, Denis Marraud, Joshua Hughes, Dairazalia Sanchez-Cortes, and Driss Khalil. Autocrime: Open multimodal platform for combating organized crime. *Forensic Science International: Digital Investigation*, 2025.

Wes McKinney. Data structures for statistical computing in python. *Proceedings of the 9th Python in Science Conference*, pages 51–56, 2010. https://doi.org/10.25080/Majora-92bf1922-00a.

Petr Motlíček, Erinc Dikici, Srikanth Madikeri, Pradeep Rangappa, Miroslav Jánošík, Gerhard Backfried, Dorothea Thomas-Aniola, Maximilian Schürz, Johan Rohdin, Petr Schwarz, Marek Kováč, Kvetoslav Malý, Dominik Boboš, Mathias Leibiger, Costas Kalogiros, Andreas Alexopoulos, Daniel Kudenko, Zahra Ahmadi, Hoang H. Nguyen, Aravind Krishnan, Dawei Zhu, Dietrich Klakow, Maria Jofre, Francesco Calderoni, Denis Marraud, Nikolaos Koutras, Nikos Nikolau, Christiana Aposkiti, Panagiotis Douris, Konstantinos Gkountas, Eleni Sergidou, Wauter Bosma, Joshua Hughes, and Hellenic Police Team. Roxsd: The roxanne multimodal and simulated dataset for advancing criminal investigations. In *Odyssey 2024: The Speaker and Language Recognition Workshop*, 2024.

Giancarlo Perrone, Jose Unpingco, and Haw-minn Lu. Network visualizations with pyvis and visjs. In *Proceedings of the 19th Python in Science Conference (SciPy 2020)*, pages 58–65. SciPy, 2020. https://doi.org/10.25080/Majora-342d178e-008. URL `https://proceedings.scipy.org/articles/Majora-342d178e-008.pdf`.

Plotly Technologies Inc. Dash Cytoscape: A Component Library for Interactive Network Visualizations in Python, 2024. URL `https://dash.plotly.com/cytoscape`.

Pradeep Rangappa, Amanda Muscat, Alejandra Sanchez Lara, Petr Motlicek, Michaela Antonopoulou, Ioannis Fourfouris, Antonios Skarlatos, Nikos Avgerinos, Manolis Tsangaris, and Kasia Kostka. Detecting criminal networks via

non-content communication data analysis techniques from the tracy project. *15th EAI International Conference on Digital Forensics & Cyber Crime (EAI ICDF2C)*, October 2024. https://doi.org/10.1007/978-3-031-89363-6$_2$0. URL `https://doi.org/10.1007/978-3-031-89363-6_20`.

ROXANNE Consortium. Roxanne platform overview. `https://www.roxanne-euproject.org/platform`.

spaCy. spacy: Industrial-strength natural language processing in python. `https://spacy.io`, 2020.

Jiao Sun, Qixin Zhu, Zhifei Liu, Xin Liu, Jihae Lee, Lei Shi, Zhigang Su, Ling Huang, and Wei Xu. Fraudvis: Understanding unsupervised fraud detection algorithms. In *2018 IEEE Pacific Visualization Symposium (PacificVis)*, pages 160–169. IEEE, 2018. https://doi.org/10.1109/PacificVis.2018.00029. URL `https://sunjiao123sun.github.io/2018/FraudVis/FraudVis.pdf`.