Enhancing Cybersecurity Event Networking with Semantic Knowledge Graphs

by Dinis Cruz and ChatGPT Deep Research, 2025/04/06

Problem Statement: The Serendipity Dilemma in Event Networking

Cybersecurity conferences and trade events thrive on the connections made between attendees, vendors, and presenters. Yet too often, who meets whom is left to chance. Even as attendee networking expectations rise (Creating Planned Serendipity For Your Conference Success), many conferences still rely on random, *"luck-of-the-draw"* encounters to spark relationships. An attendee might stumble into a future business partner over coffee, or a startup might luckily sit next to its ideal investor at lunch. While serendipitous meetings can be wonderful, the current approach leaves countless opportunities unrealized.

Organizers have begun to recognize this inefficiency. Paddy Cosgrave, CEO of the Collision tech conference, noted that manual matchmaking (curating seating charts, pairing people by hearsay) fails to scale as events grow – *"the permutations a curator needs to consider tend towards infinity"*. In other words, with hundreds or thousands of participants, it's beyond human ability to manually ensure each person meets the most relevant contacts. The result is a networking experience driven largely by chance, where critical connections are missed simply because the right people never crossed paths.

Some forward-thinking events have tried to *"engineer serendipity."* For example, Collision hired data scientists to algorithmically pair 600 startups with investors for one-on-one meetings. Their team designed algorithms to suggest *"the most purposeful face-to-face connections"*, effectively using data to do what was once left to random encounters. The lesson is clear: relying solely on organic mingling is no longer enough. There is a better way to facilitate relationship-building – one that uses technology to augment human networking and ensure that valuable connections *do* happen. Event organizers need a method to systematically connect the right people at the right time, without losing the natural, enjoyable aspect of discovery.

Semantic Knowledge Graphs as a Solution

To fundamentally improve networking at cybersecurity events, organizers can turn to **semantic knowledge graphs** as the backbone of a recommendation and matchmaking engine. A semantic knowledge graph is a structured representation of information that captures entities (people, companies, products, topics, etc.) and the meaningful relationships between them. In essence, it's a graph-based data model that acts as a living map of how everyone and everything at the event is connected. By encoding attendees, vendors, presenters, and their interests or offerings into a knowledge graph, we gain a holistic, queryable view of the event's "social network." This enables the event platform to reason about connections and surface non-obvious relationship insights that would be invisible to manual methods.

Think of it like a dynamic **"who-knows-what-and-who"** map. Each attendee could be a node linked to topics they're interested in, skills they possess, and companies they represent. Vendors are nodes connected to the products or solutions they offer and the technologies they relate to. Presenters connect to the session topics they'll cover and the expertise areas in their bios. All of these nodes interlink where there are commonalities – for example, a presenter's talk on zero-trust architecture would link to the "Zero Trust" topic node, which might also connect to several attendees who expressed interest in zero-trust, and to a vendor whose product provides zero-trust network solutions. The graph thus becomes a rich web of semantic relationships: *who* is associated with *what*, and *how* those relate to others.

Crucially, a semantic knowledge graph isn't just a static database – it's imbued with domain meaning. That means the connections aren't mere keywords but have context. For instance, the graph can distinguish that *Alice works for Company X*, *Bob is interested in Zero Trust*, *Company X offers a cloud security product*, and *Bob's interest in Zero Trust is a subset of cloud security*. These nuanced relationships allow more intelligent matching than simple tag matching. A well-constructed knowledge graph can infer links via shared attributes or broader topics. It provides a way to **"connect the dots"** across the diverse data an event collects – attendee registrations, speaker bios, vendor brochures, session abstracts, and more – in a unified model.

(Establishing Provenance and Deterministic Behaviour in an LLM-Powered News Feed (first MyFeeds.ai MVP)) *A visualization of a semantic knowledge graph generated from 50 news articles (553 nodes, 1451 edges). Even with just titles and summaries, the graph forms a dense web of interconnected entities, illustrating how much structured information can be derived from raw text. In an event context, a similar graph could link attendees, companies, and topics to reveal hidden connections.*

With such a graph in place, an event can move from **relying on serendipity to engineering serendipity**. Instead of hoping the right people happen to meet, the system can proactively suggest connections based on data. The knowledge graph approach enables what conference consultants have dubbed "*planned serendipity*" – deliberately increasing the likelihood of valuable discoveries. By mapping out relationships ahead of time, we create more opportunities for chance-like encounters that are actually guided by insight. This doesn't make networking feel forced or artificial; rather, it augments the natural flow of the event by nudging together those who have meaningful reasons to connect. In short, semantic graphs provide a way to fundamentally change how conferences facilitate networking (Data-Driven Meetings: Is Your Association Ready? | Associations Now): from a passive, luck-driven process to an active, data-driven one that remains organic in experience but optimized in outcome.

Building the Knowledge Graph with AI and Participant Collaboration

How do we construct such a comprehensive semantic knowledge graph for an event? The key is to leverage the wealth of information participants already provide – and to do it in a scalable, collaborative way. **Large language models (LLMs)** can serve as powerful engines to convert the unstructured text that attendees, speakers, and vendors submit (bios, session descriptions, product info) into structured graph data. At the same time, human participants must remain in the loop to guide and refine the graph, ensuring it stays accurate and relevant. The process is a collaborative dance between AI-driven automation and user-driven curation.

1. Data Ingestion from Participants: Early in the event lifecycle, organizers gather data: attendee registration forms often include job titles, interests or objectives for the event; speakers provide biographies and talk abstracts; vendors submit company profiles and product descriptions. This is rich raw material but typically lives in plain text. Historically, little of this text is used beyond printing in a conference booklet. Here is where we change that – by feeding this information into an LLM for semantic analysis.

2. LLM-Based Semantic Extraction: Using an LLM, we can semantically analyze each piece of text and extract the key entities and relationships expressed. For example, suppose an attendee's bio says: *"Jane Doe is a cloud security engineer at Acme Corp with a focus on IoT device security and interested in zero-trust networking."* An LLM can parse that and produce structured data such as: `Person: Jane Doe; Role: Cloud Security Engineer; Company: Acme Corp; Expertise: IoT Security; Interest: Zero Trust`. Similarly, a vendor description like "SecureHome Corp – offering AI-driven IoT intrusion detection systems" yields nodes for `Company: SecureHome Corp; Product: IoT intrusion detection; Technology: AI; Domain: IoT Security`. Each speaker's session abstract can be distilled into the topics covered, problems addressed, etc. In technical terms, the LLM is converting free text into a JSON or graph representation of entities (nodes) and their relations (edges). Recent implementations have demonstrated the efficacy of this approach: for instance, the MyFeeds project used LLMs to convert RSS news articles into knowledge graph entries, *"extracting the who, what, and how"* of each story and encoding those facts into a graph structure. In our context, the "story" is each participant's profile or content piece.

Notably, modern LLM APIs support **structured output** formats, meaning we can directly get a well-formed JSON object of extracted data from the model (Building Semantic Knowledge Graphs with LLMs: Inside MyFeeds.ai's Multi-Phase Architecture). This greatly streamlines turning LLM output into graph data. Each attendee, vendor, and presenter ends up represented as a collection of nodes and edges in the graph – for example, Jane Doe's node connects to nodes for *Acme Corp*, *Cloud Security*, *IoT Security*, and *Zero Trust*. Every piece of event data yields its own mini-graph of entities which can be merged into the overall event knowledge graph. By automating this step with AI, we solve the scale problem: whether there are 100 or 10,000 attendees, the heavy lifting of initial data processing is handled by the LLM. As Dinis Cruz observed in building such pipelines, this approach *"let the LLM pick the best relationships"* latent in the text, even without a predefined ontology, and it did *"a pretty good job"* of identifying meaningful links. All extracted data is stored in a graph database, using an open format so it can be easily queried. (For example, an open-source tool called MGraph-DB was used in the MyFeeds project to take the LLM's JSON output and seamlessly merge it as nodes and edges in a graph database. Such technology could be repurposed here to build the event graph in real-time as data comes in.)

3. Participant Collaboration and Curation: While AI automates the extraction, participants should have visibility and input into their profile in the knowledge graph. A collaborative approach might invite each attendee to review the "tags" or key points that the system has derived about them. For instance, Jane might log into the event portal and see that the system tagged her with *Cloud Security*, *IoT Security*, *Zero Trust*, *Acme Corp*, etc., based on her bio. If something is missing or inaccurate, she can adjust it – maybe she also wants to add *SCADA/ICS Security* as an interest that wasn't explicitly in her bio. In this way, the knowledge graph is enriched both by AI **and** by direct user input. This concept has precedent: the PoolParty EventAdvisor, a semantic matchmaking tool used at a semantics conference, automatically tagged attendee profiles against a knowledge graph of skills, **and then allowed attendees to add additional interests as new tags** to refine their profile (Semantic Matchmaking in Enterprise Environments - PoolParty Semantic). Those user-added tags became part of their "attendee footprint" in the graph. We can mirror this method – use AI to propose a baseline profile, then let the participant tweak or augment it.

This human-in-the-loop curation drastically improves quality. It combines the scale of AI with the judgment of domain experts (in this case, each person is the expert of their own interests). Research in ontology learning emphasizes that such **"semi-automatic"** approaches yield the best results: algorithms suggest candidates and human experts validate them (From Top-Down to Organic Evolving Graphs, Ontologies, and Taxonomies - Dinis

Cruz - Documents and Research). By validating and refining the LLM's output, participants ensure the graph truly reflects meaningful relationships and not AI hallucinations or irrelevant data. Furthermore, having users involved increases trust – they know how they're represented and can control it, alleviating privacy or misrepresentation concerns.

4. Unified Graph Construction: After extraction and user curation, the individual pieces (attendee sub-graphs, speaker sub-graphs, etc.) are merged into one unified knowledge graph for the event. This graph will have node types like **Person**, **Company**, **Product**, **Topic/Interest**, **Session**, etc., with edges denoting relations such as *works at*, *interested in*, *expert in*, *offers*, *talks about*. The merging process can be automated by the graph database, since all data is structured. Any overlapping entities (e.g., "Zero Trust" as a topic appears in multiple profiles) are reconciled to a single node for that topic, linking to all relevant people. The end result is an event knowledge graph that is both **comprehensive** and **ready for query**. In practical terms, this might be updated continuously up to and during the event as new data comes in (last-minute signups, changes in interests, etc.). Modern graph databases or even serverless graph platforms (like the aforementioned MGraph-DB) can handle merging these streams of JSON data in near-real-time . By the time the event kicks off, the organizers have at their fingertips a machine-readable map of the community and content at their event.

In summary, building the graph involves a pipeline of LLM-powered information extraction and human-guided refinement. It's a **collaborative knowledge engineering** effort where every participant's input (explicit or through provided text) helps shape an accurate representation of the conference's knowledge network. And importantly, this process is feasible today: tools and techniques to do it have been proven in analogous scenarios. For example, the MyFeeds.ai project demonstrated a multi-phase LLM pipeline that transforms raw text into a knowledge graph and beyond in a controlled, verifiable way. We can adopt a similar multi-stage approach for event data, ensuring each step (extraction, graph building, etc.) produces structured outputs that can be reviewed and debugged, rather than doing everything in a black-box. The next section will illustrate how, once this graph is built, we leverage it to supercharge the event experience with intelligent recommendations.

Connecting the Dots: From Graph to Guided Interactions

Once the semantic knowledge graph of the event is in place, it becomes a powerful engine for driving intelligent recommendations and structured networking opportunities. The system can now **interconnect** attendees, vendors, and presenters by traversing the graph to find relevant relationships. In practice, this means the event platform can suggest: "You should meet Alice, because you both care about automotive IoT security and she has expertise in an area you're exploring," or "These four people and one vendor all align around zero-trust architectures – how about we arrange a small group discussion for them?” By using the knowledge graph, such suggestions factor in multiple data points and semantic connections, going well beyond simplistic matchmaking. Let’s break down how the graph can be used to enhance different types of interactions:

- **Attendee-to-Attendee Matchmaking:** The most straightforward use-case is recommending one attendee to another. The graph can be queried for people who share significant common nodes (e.g. common interests, similar research areas) or complementary nodes (e.g. one has a problem, another has experience solving that problem). For example, if Bob indicated interest in “ransomware defense” and Carol has years of experience in incident response (and perhaps even worked on notable ransomware cases), the system can flag this as a high-value connection. Unlike basic keyword matching, the semantic graph can account for hierarchies and synonyms – if Bob wrote “malware” and Carol wrote “ransomware,” the system still links them because it knows ransomware is a type of malware. In the PoolParty EventAdvisor scenario, their matching algorithm **“explores the knowledge graph to link attendees with other people and opportunities”**, taking into account semantic relations defined by experts. This means attendees can be matched on broader skill-to-need relationships, not just identical interests. In our case, an attendee looking to learn about cloud threat hunting could be matched with another attendee who has that skill or with a speaker who covered it in a session. The knowledge graph provides the substrate to discover these overlaps and intersections automatically. Each connection suggestion can come with an explanation drawn from the graph, e.g., *“Suggested because you both listed ‘ICS/SCADA security’ as an interest.”* This not only helps break the ice when they meet (they immediately know what common ground to start on), but also builds trust in the system’s recommendations. We can make these introductions via the event app: a notification or a suggested meet-up in the schedule.
- **Attendee-to-Presenter Connections:** Often, attendees desperately want to talk with presenters (e.g., to ask follow-up questions or seek advice), but outside of brief Q&A sessions, the opportunity is missed. With the graph, we can proactively facilitate this. For instance, if a participant’s profile closely matches the topic of a presenter's talk or paper, the system can suggest the attendee attend that session (if they weren’t already planning to) and afterwards arrange a brief meeting or introduction with the presenter. If Dr. Alice is giving a talk on zero-trust implementation and Bob’s profile shows zero-trust is one of his major projects, the app could notify Bob: “The upcoming talk on *Zero Trust in Enterprise* aligns with your interests. Would you like to join a small roundtable with the speaker afterwards?” The graph knows Alice→talks about→Zero Trust and Bob→interested in→Zero Trust, thus it can bridge that connection. This could be done for panelists and workshop leads as well. Essentially it treats sessions as content nodes in the graph linking presenters and topics to attendees. By suggesting these targeted interactions, events can greatly increase presenter-attendee engagement beyond the podium.
- **Attendee-to-Vendor (Attendee-to-Exhibitor) Matching:** From an expo perspective, attendees often struggle to figure out which vendor booths to prioritize, and vendors aim to find the most interested prospects. A semantic graph can dramatically improve expo ROI for both sides. Each

vendor has nodes for the product categories or problems they solve. Each attendee has nodes for the solutions they seek or challenges they face. Matching these in the graph yields tailored recommendations: *"You should visit SecureHome Corp at Booth #12; they offer IoT intrusion detection, which aligns with your interest in IoT device security."* Likewise, vendors could get a list of the top X attendees whose profiles suggest a need for their solution (without exposing any personal data – the system can facilitate a meeting invite on behalf of the attendee's expressed interest). This goes beyond generic labels like "network security" by using detailed descriptions. If an attendee's profile says "interested in automated pentesting tools for cloud," the system might connect that to a vendor offering a cloud pentesting automation product, even if the wording differs. By traversing connections in the graph (cloud → security → pentesting), it identifies the relevance. Such matchmaking ensures that **relevant buyers and sellers find each other** more efficiently than hoping an attendee wanders by the right booth. It can also be used to group attendees on guided expo tours by interest area. For example, a "DevSecOps Tools Tour" can be formed if the graph finds 10 attendees all interested in DevSecOps; the system can then route them together through the expo to those vendors, creating a mini networking group in the process.

- **Group Discussions and "Birds of a Feather" Sessions:** The knowledge graph can uncover clusters of people who share a common interest or goal, which is an opportunity to create ad-hoc group interactions. Imagine the graph reveals that 15 attendees and 2 presenters all have a strong link to the topic *"AI in Threat Intelligence."* This might prompt the organizers to set up a *"Birds of a Feather: AI for Threat Intel"* roundtable or lunch table on the fly. Invites can be sent to those graph-connected individuals suggesting they gather to discuss that topic. Because the graph can handle many simultaneous topics, you could personalize the networking: instead of one generic networking reception, you have **themed micro-networking** meetups aligning with what people actually care about. This targeted approach increases the chances that conversations at those meetups will be deeply engaging. It's essentially recreating the hallway coffee-break serendipity, but in a planned way: you're likely to be surrounded by people who *speak the same language* on a topic. Attendees can of course opt in or out, but the ones who join will find kindred spirits waiting. This concept was foreshadowed by event best-practices; for instance, conference advisors have suggested grouping like-minded people in small seating clusters to *"spark impromptu conversations"*. The knowledge graph just supercharges and scales that idea – it identifies which minds are alike and helps bring them together effortlessly.
- **Seating Arrangements and Social Events:** For more structured interactions like seated dinners, workshops, or even on those famed *pub crawls* many conferences organize, the graph can be used to algorithmically optimize the mix of participants. Instead of random table assignments or people sitting only with colleagues they know, the system could create seating charts that maximize valuable encounters. For instance, at a 10-person dinner table, the algorithm might place a mix of roles and common interests: perhaps two CISOs interested in cloud security, seated with a cloud security product vendor, a researcher who just published on cloud vulnerabilities, and a couple of practitioners who are looking to implement cloud security solutions. Everyone at that table has overlapping interests as per the graph, giving them plenty of shared ground to start

conversations. This was exactly the kind of scenario engineered by Collision's algorithms – they even accounted for who to place together on casual outings like a pub crawl, aiming to put people together who would *"benefit from being in a meeting or on a pub crawl together"*. The outcome is a form of planned seating that feels natural (people still meet *new* folks, just highly relevant ones). Organizers no longer have to fret over manually curating these plans; they can trust the graph-driven system to produce suggestions, which they can then approve and implement. By the end of the event, attendees often marvel at the *"coincidence"* of meeting just the right person at dinner or sitting next to someone with mutual interests – not realizing it was an orchestrated serendipity courtesy of the knowledge graph.

All of these use cases hinge on one ability: the system can **query the knowledge graph for relationships of interest**. Whether it's finding people who share an attribute, or those who complement each other, or those connected by a common third node (like a topic or company), the graph makes it computationally tractable. This is where semantic richness shines. Because we're not just matching text strings, but actual concepts and relationships, the recommendations can consider the context. A matching algorithm backed by a graph can say "Alice knows X and Bob needs X" as a reason to connect them, which is a level of reasoning above plain keyword matching. In the PoolParty example, one relation they used was connecting **skills to job roles** to find matches. We can incorporate similar expert-defined or learned relations specific to cybersecurity events – e.g., connect *"interest: compliance"* with *"role: CISO"*, on the logic that CISOs care about compliance, thereby clustering those folks.

Finally, an important aspect of using the graph for suggestions is **explainability**. Each recommendation can be justified by tracing the path in the graph that led to it. This is crucial for user trust. If the system suggests I meet someone, it should tell me why. Fortunately, the knowledge graph provides a clear chain of reasoning: e.g., *"Recommended connection because you both marked interest in 'ICS Security' and you both work in the energy sector"*. In technical terms, we can generate an explanation by extracting the intermediate node (or nodes) that link the two people. If person A and B are connected through three different topics in the graph, that could be explicitly listed. This approach is akin to what MyFeeds does in content recommendations: every suggested news article to a user came with a provenance trail showing which topics overlapped between the article and the user's profile. If someone asked "why was I matched with this person?", the system can point to the graph link – *"you're both interested in X"* – turning what could be an opaque AI decision into an understandable suggestion. This transparency will make attendees (and organizers) more comfortable with the recommendations and more likely to act on them.

In summary, once the event's semantic knowledge graph is built, it can be harnessed in myriad ways to enhance interactions: - One-on-one meetings that would have unlikely happened by chance. - Themed group conversations that tap into shared interests. - Smarter introductions to relevant experts, clients, or partners. - Optimized seating and networking plans that take the guesswork out of who should mix with whom.

All of it is geared toward **maximizing relevant relationship-building**. Instead of hoping for magic, the organizers effectively give fate a gentle nudge using data and AI. Attendees still experience the thrill of a “random” great connection or a discovery in a session, but behind the scenes the graph is stacking the deck in favor of those happy accidents. This leads to richer networking outcomes: more contacts exchanged, more follow-up conversations post-event, and ultimately a stronger community of cybersecurity professionals who found value in each other.

Continuous Improvement: Feedback Loops for a Smarter Graph

Building and using the semantic graph is not a one-off task – it’s an ongoing process. Just as relationships in the real world evolve, our knowledge graph should continuously improve and update. To achieve that, we establish feedback loops that learn from each event interaction and from user feedback. This ensures that over time, the recommendations get smarter and the underlying graph becomes an even more accurate reflection of reality. In a sense, the knowledge graph becomes a *living*, evolving asset that grows with each conference.

1. Real-Time Feedback During the Event: As the event unfolds, we can collect data on which suggestions were useful and which weren’t. The event app might allow attendees to mark a recommended connection as “met” or to skip it. We could send quick surveys: *“Did you find your meeting with Alice valuable?”* If an attendee indicates a connection was very fruitful, that’s positive reinforcement that those two nodes in the graph indeed should be closely linked. The system can strengthen that link (or add a new direct connection “Alice ← met → Bob at Conference X”). If many people ignore a certain type of recommendation, that might signal a false assumption in the graph or an interest that was tagged incorrectly. For example, if none of the recommended matches based on “blockchain security” actually meet, perhaps that topic was too broad or mischaracterized – participants might refine their interests or we adjust our matching criteria. Essentially, **user interactions serve as implicit feedback** to validate or refute the graph’s connections. Modern systems often do this: in semantic search, if users keep skipping results for a query, it signals the need to tweak the ontology or mapping. Similarly, our event platform observes interactions to refine its understanding. We maintain a feedback loop: recommendations → user action/inaction → graph update.

One could imagine visualization of the graph during the event with some analytics: which interest areas are actually sparking the most connections? If a particular anticipated hot topic isn’t actually bringing people together, maybe our understanding of it needs adjustment. Conversely, if unexpected clusters of networking form (say a lot of folks from different stated interests all end up talking about a new emergent issue), that insight can be fed back in – perhaps by adding a new node to the graph for that emergent topic and linking those people to it. The system might even prompt an organizer: “It looks like ‘secure AI coding’ is trending in conversations – consider adding a discussion topic.” This way, the graph adapts dynamically, guided by actual behavior.

2. Post-Event Analysis and Iteration: After the conference, organizers can perform a thorough analysis using the knowledge graph as a memory of what happened. They can evaluate which suggested matches occurred and led to ongoing relationships (perhaps tracked via connection requests on the platform), and which suggestions were declined. They can incorporate explicit feedback from attendees: many events send post-event surveys asking “What was your favorite connection made?” or “Did you feel you met the people you wanted to meet?”. These responses can be mapped back to the graph. If attendees identify connections that were great but not suggested by the system, we investigate how we missed them – maybe a connection path existed in the graph but with too many hops or too low weight, suggesting we adjust our recommendation algorithm thresholds.

This continuous improvement is akin to what Collision’s team described: they treat each year’s event as a new “model” or iteration, learning from the last. *“No gathering is the same the following year. Each year is a new model or operating system,”* wrote Cosgrave, highlighting how they fix and update their approach annually based on what they observed. We can formalize this: after each event, we update the ontology/taxonomy that our graph uses, perhaps adding new categories for interests that emerged, merging or splitting nodes if we found attendees grouped differently than expected. It becomes an evolving **knowledge base of the community**. If this is a recurring cybersecurity conference, the graph built this year can seed the next year’s graph. Returning attendees will carry over their refined profiles (updated with any new interests or roles), and new attendees can be onboarded into the existing ontology, which has been battle-tested by prior feedback. Over multiple iterations, the ontology of topics and relationship types might organically evolve to fit the community – a shift from a top-down static list of interests to a more organic, crowd-sourced structure. As one whitepaper noted, *“human-in-the-loop governance”* combined with open contributions keeps an evolving knowledge base credible and useful. In our case, that means the organizers remain curators of the graph’s evolution (maybe convening an advisory board to review big changes), while letting attendee data and feedback drive many of the refinements.

3. Ongoing Human Oversight: To maintain quality and trust, we incorporate periodic human review of the knowledge graph and the recommendation rules. This could involve domain experts (perhaps community volunteers or a committee of organizers and frequent attendees) who look at how the graph is growing. They might prune out spurious nodes that crept in via AI extraction errors, or rename nodes for clarity (e.g., unify “AI security” and “artificial intelligence security” into one). This is analogous to how Wikipedia thrives: open contributions but with vigilant editors and consensus building. Human oversight guards against the graph devolving into an inaccurate mess, especially as we incorporate more and more data. For example, if left unchecked, an LLM might overzealously create a separate interest node for “cloud-security” vs “cloud security” due to a hyphen;

human curators can spot such duplicates and merge them. Or they might enforce that acronyms and full terms are linked (e.g., “IoT” and “Internet of Things” refer to the same concept). These governance practices ensure the graph’s semantic structure remains coherent and high-quality. The feedback loop thus isn’t just algorithmic; it’s also organizational, where insights from the event team and participants lead to deliberate improvements.

4. Privacy and Participant Trust: Continuous improvement also means continuously earning participants’ trust. We should treat participant data with care – the feedback loop can include getting consent for using interactions as learning data. Transparency is key: if attendees know that their input (like skipping a suggestion) helps make the system better, they’ll be more inclined to provide it. And as mentioned, giving explanations for suggestions goes a long way. If a suggestion wasn’t on point, the user seeing *why* it was made (perhaps a wrong assumption about their interest) allows them to correct the profile. This user correction is itself a direct feedback that improves the graph. In essence, every interaction – whether accepting a meeting, declining it, editing one’s tags, or post-event surveys – feeds into a loop of learning. Over time, this process transforms the knowledge graph into a very accurate model of the community’s interests and relationships. It becomes a **living knowledge ecosystem**, continuously refined, rather than a static data snapshot.

In technical terms, we might even use an AI to periodically retrain or re-run analysis on new data. If between events a whole new subfield of cybersecurity becomes hot (say, “quantum encryption”), we can feed new articles or discussions to an LLM to expand the ontology so that next event the graph is up-to-date. The AI can propose new nodes and relations reflecting the latest trends, which humans then vet. This way, the knowledge graph remains a **current, accurate representation of the domain and the attendee community**.

By implementing these feedback loops, event organizers ensure that the investment in building a semantic graph yields increasing returns. The more the system is used, the smarter it gets. Attendees will notice that each year (or each event iteration) the recommendations feel more “on the nose” for what they’re looking for, and the event’s networking keeps improving. In the end, the continuous refinement process will make the semantic matchmaking system an integral, trusted part of the event – something attendees even look forward to using, because it demonstrably helps them connect with the right people. It turns the conference from a one-time gathering into part of a sustained knowledge network that lives on and evolves.

Implementation in Practice: Lessons from MyFeeds.ai and Open-Source Tools

The vision outlined above – from LLM-driven graph construction to intelligent recommendations and feedback loops – can sound complex. However, recent advancements and prototype projects show that it's entirely achievable with today's technology. In fact, components of this approach have already been implemented in other domains. This section looks at a practical example and the tools available, to give event organizers a sense of how they could build or acquire such a system for their own conferences.

One relevant case study comes from **Dinis Cruz's MyFeeds.ai project**, which tackled a similar challenge in a different context: personalized cybersecurity news feeds. The system needed to match news articles to readers' interests, which, as it turns out, involves building and connecting knowledge graphs not unlike those for event participants. The MyFeeds pipeline offers a blueprint for how LLMs and graphs work in concert in a multi-stage process. Here's a quick recap of how it worked, and how those ideas translate to our event scenario:

- **Stage 1: Entity & Relationship Extraction.** MyFeeds would take an incoming piece of text (a news article) and use an LLM to extract the key entities and their relationships, outputting a structured JSON graph of the article's content. In our case, this corresponds to processing an attendee bio or vendor description into a graph snippet. The emphasis is on structured output – the LLM isn't just summarizing in prose; it's filling a predefined schema with entities like "Person", "Skill", "Company" etc. This structured approach is crucial for traceability and consistency.
- **Stage 2: Persona Graph Construction.** MyFeeds next constructed a graph representing the user's interests (their persona) via LLM, also as JSON. This is analogous to building an attendee's interest graph or a profile graph for each participant. In events, we largely get persona info from the user's input directly, but an LLM can still help by expanding or enriching it (for instance, if someone says "I work on OT security", the system might infer related interests like "ICS" or "critical infrastructure" nodes to add to their persona graph). In MyFeeds, this persona graph was described as a *"semantic fingerprint"* of what the user cares about. For our purposes, each attendee's profile graph serves that role.
- **Stage 3: Relevance Mapping.** In MyFeeds, given an article graph and a persona graph, an LLM then compared them to find intersections – essentially answering "why would this article be relevant to this person?" and outputting the connections as data. For event matchmaking, this step is akin to comparing two attendee graphs (or an attendee and a vendor's graph, etc.) to find common nodes that would justify a meeting. We might implement this as a graph query rather than an LLM in some cases, but an LLM could augment by finding less direct connections too. The

key is the output is structured: a list of the overlapping topics or complementary links between two entities. This is exactly what we need to produce human-readable reasons for recommendations.

- **Stage 4: Recommendation (Summary) Generation.** Finally, MyFeeds used an LLM to generate the personalized news summary for the user, incorporating the relevant points identified. In our scenario, this could be the step where the system formulates the actual suggestion message for a user: taking the raw connection data from Stage 3 and writing a friendly invitation like, *"We suggest you meet Alice from Acme Corp – you both work on IoT Security and she's tackling the same challenges you mentioned in OT environments."* The LLM can fill in the template and tone, but it's constrained by the facts identified (to ensure correctness). This is more of a nice-to-have (one could also use templating without AI), but it adds a conversational tone to the recommendations which can improve engagement.

What's important about the MyFeeds implementation is how it stressed **provenance, explainability, and determinism** throughout. Each stage produced a verifiable artifact (JSON data) that could be inspected. If something went wrong, they could pinpoint which stage misinterpreted the data. For an event system, this is equally important: we want to be able to troubleshoot why a certain match was made or why someone's profile was parsed a certain way. By breaking the process into clear steps and outputs, we maintain control. It turns the matchmaking AI from a mysterious black box into a transparent assistant. As Dinis Cruz noted, capturing intermediate results as data provides a *"provenance trail"* that shows why a particular recommendation was made. For example, if a participant asks "Why was I recommended to join the ICS roundtable?", the system can trace: *"Your profile mentioned working with SCADA systems, and the roundtable was for 'ICS Security' which is related – see this connection in the graph."* This aligns perfectly with the needs of professional events where participants might be curious (or skeptical) about algorithmic suggestions. Transparency builds trust.

In terms of **tools and open-source technology**, much of what's needed to build this already exists:

- **Graph Database / Knowledge Graph Platform:** We need a place to store and query the semantic graph. This could be a traditional graph database like Neo4j or AWS Neptune, or newer solutions. In MyFeeds, Dinis built **MGraph-DB**, a serverless graph database optimized for merging LLM outputs. MGraph-DB is open-source and designed to easily take JSON from LLMs and treat them as graph entries, which could be a strong starting point for an event graph implementation. It allows for memory-first operations, meaning the graph can be manipulated on-the-fly in code, which suits our iterative building process. Alternatively, one could use RDF triplestores or property graph DBs with a defined ontology for the event (e.g., using RDF/OWL to define schema for Person, Company, etc.). The choice might depend on scale and familiarity, but the key is that the database should handle dynamic schema growth since our ontology will evolve.

- **LLM Integration:** There are various ways to integrate LLMs. OpenAI's GPT-4 or similar large models can be used with *function calling* or *structured output* enabled, to directly get JSON results. This was demonstrated in practice – MyFeeds used OpenAI's structured response schemas to ensure the LLM output conformed to the expected classes (Building Semantic Knowledge Graphs with LLMs: Inside MyFeeds.ai's Multi-Phase Architecture). We can define a schema for our event data extraction (for example, a class `AttendeeProfile` with fields like name, title, interests: [list], company, etc., and relations) and have the LLM populate it. For an open-source approach, there are also libraries like Hugging Face transformers or LangChain that can be set up to perform extraction with smaller models or even rule-based NLP for certain tasks. The advantage of LLMs is their flexibility and ability to understand context (they can handle messy input from participants gracefully), but for critical fields like names or companies, simpler parsing might complement to ensure accuracy.
- **User Interface for Collaboration:** To involve participants in curating their profile, a user-friendly interface is needed. This could be built into a conference mobile app or web portal. For example, after registration, an attendee could be prompted to *"Review your profile keywords"*. This UI would display the tags extracted and allow adding/removing. Implementing this is straightforward with modern web frameworks. The key is that changes in the UI feed back into the graph database (e.g., adding a tag creates a new node+edge for that person). Technologies like GraphQL could be useful here, providing a layer to query and update the graph via API for the front-end.
- **Recommendation Engine:** On the back end, once the graph is built, we need to run queries or algorithms to generate the suggestions. This might involve graph algorithms such as community detection (to find clusters for group sessions), similarity measures (to rank one-on-one match likelihood), or simple rule-based queries (find all pairs of attendees who share at least two interest nodes, for example). Many graph databases have built-in algorithm libraries. We could also use an LLM at query time – for instance, feeding it a subgraph and asking it to suggest an interesting connection. However, for efficiency and determinism, algorithmic approaches may suffice. One could frame it as a recommendation problem and use collaborative filtering or link prediction techniques on the graph: treat it like a recommendation of "people you may want to meet" analogous to "people you may know" on social networks, but leveraging the rich semantic links. Open-source frameworks in Python like NetworkX or Neo4j's GDS can help with prototyping these algorithms.
- **Integration with Event Schedules:** The suggestions we generate need to map onto actual times/places in the event agenda. This requires integration with the scheduling system. For example, if the graph suggests Alice and Bob should meet, the system needs to check their schedules for a mutual free slot or perhaps schedule a new meeting for them (some conference platforms allow you to book meetups). Implementing this might involve an optimization component that schedules numerous recommended meetings without conflict – essentially solving a matching problem. There are scheduling libraries or one could formulate it as a constraint satisfaction problem. In simpler terms, the system could propose

a time like a specific “networking break” when both are free. Many events also have a dedicated matchmaking session or “office hours” times for vendors/presenters; those slots can be filled with the highest priority matches first.

- **Security and Privacy Considerations:** Given it’s a cybersecurity event, it’s worth noting that any system handling personal data and making connections should do so securely. The knowledge graph might contain sensitive information (e.g., an attendee’s interest in finding a new job – which they might not want their current employer to see). Access control is therefore important: not everyone should see the whole graph, only the system and maybe the organizers at an aggregate level. The recommendations to users should reveal just enough info to entice the connection without breaching privacy. For instance, the platform might internally know that Bob wants to discuss “job opportunities” but it wouldn’t broadcast that; it might instead facilitate anonymous reach-out to relevant vendors or mentors. All these can be handled with proper data governance in the implementation phase.

The good news is that none of this requires reinventing the wheel. The work by Dinis Cruz and others shows that **open-source components can be glued together** to build such a system. MyFeeds.ai’s code (for semantic parsing, multi-step LLM calls, etc.) can be a reference. Tools like MGraph-DB are available for use or adaptation. Established ontologies like ESCO (skills classification) were used in PoolParty’s EventAdvisor; we might leverage cybersecurity-specific taxonomies (like NICE framework for skills, or MITRE ATT&CK for techniques) as a starting vocabulary for our graph and let the LLM map user text to those. This can jump-start the semantic accuracy (e.g., recognizing that “pen testing” maps to “penetration testing” concept in the ontology). In fact, PoolParty’s example showed the benefit of anchoring a graph in a well-defined schema – they built their event graph on top of the ESCO vocabulary to classify skills and interests. We could do similarly with a mix of standard and custom terms.

Finally, implementing this vision is as much about process as technology. It would be wise to pilot it on a small scale: perhaps for one track of a conference or as an opt-in beta feature for attendees who are interested. That way, organizers can fine-tune the system and demonstrate its value. As comfort grows, it could expand to be a core part of all networking activities.

The bottom line is, the pieces are in place to bring semantic knowledge graphs into the realm of event management. By stitching together AI-driven text understanding, graph databases, and user-centric design, even a modestly resourced team can stand up a working prototype. The **resulting system can transform a cybersecurity event from a traditional meet-and-greet environment into a smart, interactive experience** where every participant is empowered with personalized opportunities to connect. It’s a chance to materially improve the ROI for attendees (in knowledge and contacts gained), for vendors (in leads generated), and for presenters (in influence and feedback), all by using data that was previously going underutilized.

Conclusion: A Vision for Smarter Cybersecurity Conferences

In this white paper, we presented a forward-looking yet practical approach to enhancing relationship-building at cybersecurity events. By introducing semantic knowledge graphs into the fabric of conferences, organizers can move beyond the old paradigm of hoping the right people just happen to meet. Instead, they can actively facilitate the connections that matter – connecting attendees to peers, experts, solutions, and ideas that align with their goals.

We began by identifying the core problem: the heavy reliance on serendipity in current events, which leaves too much to chance and too little to strategy. We then outlined a solution that treats information as the connective tissue of the event. By representing attendees, vendors, presenters, and content in a unified semantic graph, we unlock the ability to search and match on meaning, not just on superficial criteria. With the help of LLMs, this graph can be constructed from the rich data participants already provide, and refined with their collaboration to ensure accuracy and buy-in.

The power of this approach lies in what the knowledge graph enables: recommendations for one-on-one meetings that would have been missed, intelligently curated group discussions on pressing topics, and even optimized seating arrangements that maximize productive encounters. All of this can happen seamlessly, embedded into the event's workflow, so that from the attendee's perspective it feels like the event just magically had *"the right people in the right place."* But as we discussed, it's not magic – it's the careful application of AI and data. And thanks to a design focused on transparency (with traceable graph-driven suggestions), the system can remain accountable and earn users' trust.

We also delved into the importance of continuous improvement. A knowledge graph is not static; it's a living model of an ever-evolving community. With feedback loops and human oversight, the graph becomes smarter over time, learning from each event and thereby making each subsequent event more valuable. This turns conferences into iterative learning experiences not just for attendees but for the organizing process itself. The event effectively develops its own memory and intelligence about what connections yield the best outcomes.

The practical implementation section showed that this isn't science fiction. The components – LLMs for text-to-graph, graph databases for storage, algorithms for matching, interfaces for user input – are at our fingertips. Projects like MyFeeds.ai demonstrate that the fusion of LLMs and knowledge graphs can be done today, and they offer patterns we can emulate. Enterprise tools like PoolParty have already applied semantic matchmaking in event settings with success. Building on open-source tools and proven techniques means organizers don't have to start from scratch. With a clear vision and the willingness to experiment, even the traditional conference can be re-imagined as a smart, adaptive networking platform.

For event organizers in the cybersecurity field, the stakes are high. The domain is vast and complex – matching the right expert with the right problem can spark innovations that make us all safer. The professionals attending your conference are pressed for time and overloaded with information; by using a semantic knowledge graph to personalize their experience, you help them cut through the noise and zero in on what (and who) will most enrich them. Vendors and solution providers get to meet genuinely interested prospects rather than making cold pitches. Presenters see their message travel further and engage deeper through follow-on discussions. The entire ecosystem benefits from a more tightly woven web of connections.

In a slightly conversational but earnest tone, imagine telling your attendees next year: *"We've implemented an intelligent networking system – an AI that learns about what you do and what you need, and it will suggest people and sessions for you to check out. It's like having a personal concierge for connections."* Most will be intrigued, some might be cautious – but as they start experiencing those *"wow, what a coincidence!"* moments engineered by the system, they'll become believers. And when they fill out the post-event survey, instead of saying "networking could have been better," they'll say "met so many great contacts, the app recommendations were on point."

This is the future we can work toward: **cybersecurity conferences that are not just events, but curated knowledge-sharing experiences powered by semantic technology.** By bridging human insight and machine intelligence, we ensure that no attendee leaves with the regret of "I wish I had met someone who could help with X." The graph will have helped make that connection. It's a future where networking is both spontaneous *and* systematically supported – where serendipity is not left to mere chance, but is proactively encouraged in all the right ways.

Ultimately, embracing semantic knowledge graphs is about enhancing the human element of events. It may seem paradoxical that high-tech AI and graphs lead to more human-to-human interaction, but that is precisely the promise. When technology handles the grunt work of analysis and matching, people are free to do what they do best: communicate, collaborate, and build relationships. The white paper you've read lays out the roadmap for making this happen. The next step is in your hands as an organizer – to pilot these ideas, learn from them, and refine them. The tools are ready, the data is waiting, and your attendees are hopeful for more meaningful engagement. By connecting the right dots, you can deliver an event experience that truly resonates and creates lasting value in the cybersecurity community. It's time to transform serendipity from a happy accident into a designed feature of your events. The technology is here; the opportunity is clear. Let's graph the future of networking, one node at a time.

Sources: The concepts and techniques discussed are informed by recent advancements in semantic AI and practical implementations in adjacent domains. Dinis Cruz's work on multi-stage LLM pipelines and semantic graphs provided a reference architecture (Establishing Provenance and Deterministic Behaviour in an LLM-Powered News Feed (first MyFeeds.ai MVP)), demonstrating how structured AI outputs can drive personalized content with full traceability. Real-world event matchmaking tools like PoolParty's EventAdvisor have shown the efficacy of using knowledge graphs and ontologies (e.g., ESCO) to connect attendees with relevant talks and people (Semantic Matchmaking in Enterprise Environments - PoolParty Semantic) . Industry experiences, such as Collision's data-driven networking approach, highlight both the need for and success of "engineering serendipity" at scale (Data-Driven Meetings: Is Your Association Ready? | Associations Now). Research on human-in-the-loop knowledge management reinforces the importance of combining algorithmic suggestions with expert curation and feedback for an evolving ontology (From Top-Down to Organic Evolving Graphs, Ontologies, and Taxonomies - Dinis Cruz - Documents and Research). This white paper synthesizes these insights into a cohesive vision tailored for cybersecurity events, aiming to inspire organizers to harness semantic technology in solving the age-old challenge of bringing the right people together. Each citation throughout the text points to specific examples and findings that back the strategies proposed, ensuring that our recommendations are grounded in evidence and real-world learnings.