Big-Data Graph Knowledge Bases for Cyber Resilience

Steven Noel, Deborah Bodeau, and Rosalie McQuaid
The MITRE Corporation
McLean, Virginia and Bedford, Massachusetts
USA

[snoel, dbodeau, rmcquaid]@mitre.org

ABSTRACT

This paper describes the application of MITRE's CyGraph *tool for proactive and reactive cyber resilience. Employing a multi-relational property graph formalism, CyGraph combines data from numerous sources to build a unified graph representation for network infrastructure, security posture, cyber threats, and mission dependencies. This forms an enterprise resilience knowledge base for remediating attack vulnerability paths and responding to intrusion events, while focusing on the protection of key cyber assets. We leverage our previous work in topological vulnerability analysis for mapping known vulnerability paths through a network, along with capabilities for mapping enterprise mission dependencies on cyber assets. We then extend this by discovering and prioritizing risky multi-step patterns among traffic flows, alerts, and vulnerabilities. Through the CyGraph resilience knowledge base, we associate risky network traffic paths with traffic filtering devices and rules allowing them, for proactive remediation and reactive mitigation. CyGraph employs NoSQL graph database technology to store and process the resilience knowledge base at scale, with domain-specific queries that uncover multi-step reachability from threats to vulnerable hosts and key cyber assets.*

1.0 INTRODUCTION

Cyber resilience is the ability to anticipate, withstand, recover from, and adapt to adverse conditions, stresses, attacks, or compromises on cyber resources [1]. Cyber resilience can be a property of an individual system, a network, a mission, a system-of-systems on which a mission depends, or an organization. Understanding risks enables systems engineers and cyber defenders to achieve an appropriate level of cyber resilience in a system or network. Risk to mission can also serve as a resilience metric [2].

Risk is a function of *likelihood* and *impact*, i.e., the likelihood that a particular adverse event will occur and the impact that the event has on some important outcome. In the context of cybersecurity, likelihood is usually decomposed to include likely *threats* against an enterprise and known *vulnerabilities* of enterprise systems [3], and sometimes other factors such as the availability of countermeasures [4] [5]. Risks can be *mitigated* by reducing either likelihood or impact severity; vulnerabilities can be *remediated* by removing them (e.g., by patching incorrect code, by removing an unused piece of code the flaws in which have not been determined) or by reducing their exposure. Mitigation and remediation can be *proactive*, based on analysis which reveals the potential for a risk to materialize, or *reactive*, based on analysis which takes specific threats into consideration in the assessment of likelihood and/or impact severity.

Because of complex interdependencies among networked systems, risks associated with individual hosts, vulnerabilities, and events should not be considered in isolation. Advanced adversaries usually expand their network presence through incremental movement. Moreover, complex mission systems and systems-of-systems are deployed across a multitude of networked cyber assets. In such contexts, both the likelihood and impact aspects of cyber risk are not determined by individual hosts, threats, vulnerabilities, or alerts. Rather, they are emergent properties of the patterns of relationships among such entities.

Graphs are an ideal representation for encoding such entities and relationships in the cyber domain. However, traditional graph formulations with entities (vertices) and relationships (edges) of a single homogeneous type lack the expressiveness required for representing the rich structures involved in analyzing cyber risk. So-called *property graphs* [6] are attributed, multi-relational graphs [7], with vertices and edges of multiple types having arbitrary key/value attributes (properties). Property graphs have the power needed for expressing a range of heterogeneous vertex and edge types, which arise from combining data from a variety of sources into a coherent unified cybersecurity graph model.

A number of software systems exist for storing and computing over property graphs, including NoSQL graph databases [8] such as Neo4j [9] [10] and JanusGraph [11], RDF stores such as Rya [12], and the Apache TinkerPop [13] graph computing framework. There have been standardization efforts for querying non-relational graph databases [14], and there is multi-vendor support for such graph query languages as Cypher [15], SPARQL [16], and Gremlin [17]. While there are some languages with imperative features, graph query languages are generally declarative [18], in which one specifies a graph query pattern to be matched, rather the giving the specific instructions accessing the data. Rather, the database implementation accesses the data based on the query declaration, allowing for implementation-specific optimizations. There is direct correspondence between a graph data model and language for querying it [19], i.e., data analysis needs to match data representation.

MITRE's *CyGraph* [20] [21] is a methodology and tool for improving network security posture, maintaining situational awareness in the face of cyberattacks, and focusing on protection of mission-critical assets. CyGraph constructs a cybersecurity *knowledge graph*, i.e., a graph representing a network's security posture, based on a cybersecurity knowledge base. It enables identification of vulnerability paths, where a *vulnerability path* is an attack path through a network in which vulnerabilities are sequentially exploited, and which can be broken by remediating a vulnerability.

Given data from various network and host sources, CyGraph leverages NoSQL graph database technology to capture the complex relationships among entities in the cybersecurity domain. It employs graph queries for identifying risky patterns with prioritization of the matched subgraph clusters. Domain-specific CyGraph Query Language (CyQL) is compiled to the query language native to the backend graph database. Employing web-based client-server architecture, CyGraph provides interactive graph visualization in the browser for navigating the results of queries. CyGraph discovers and prioritizes risky patterns among multi-step relationships in network data, and guides proactive remediation and reactive mitigation. This provides more mature cybersecurity that is informed by threats and mission needs, and founded on practical experience [22].

2.0 SYSTEM ARCHITECTURE

As shown in Figure 1, CyGraph ingests data from various sources and normalizes them. It then transforms the elements of the normalized model into a graph model specific to the cybersecurity domain. Graph queries are issued from the client front end, executed on the backend database, and the resulting query matches are visualized.

In this agile architecture, the graph model is defined by how the data sources are transformed into a property graph, rather than conforming to a predetermined schema. Model extensions are simply the creation of additional of nodes, relationships, and properties in the property graph model, and require no schema changes or other database renormalizing. CyGraph currently supports two options for backend data storage and query processing:

- Neo4j graph database [9] [10] with normalized data in Elasticsearch [23] [24].
- Apache Rya [12] RDF store with normalized data in Apache Accumulo [25].

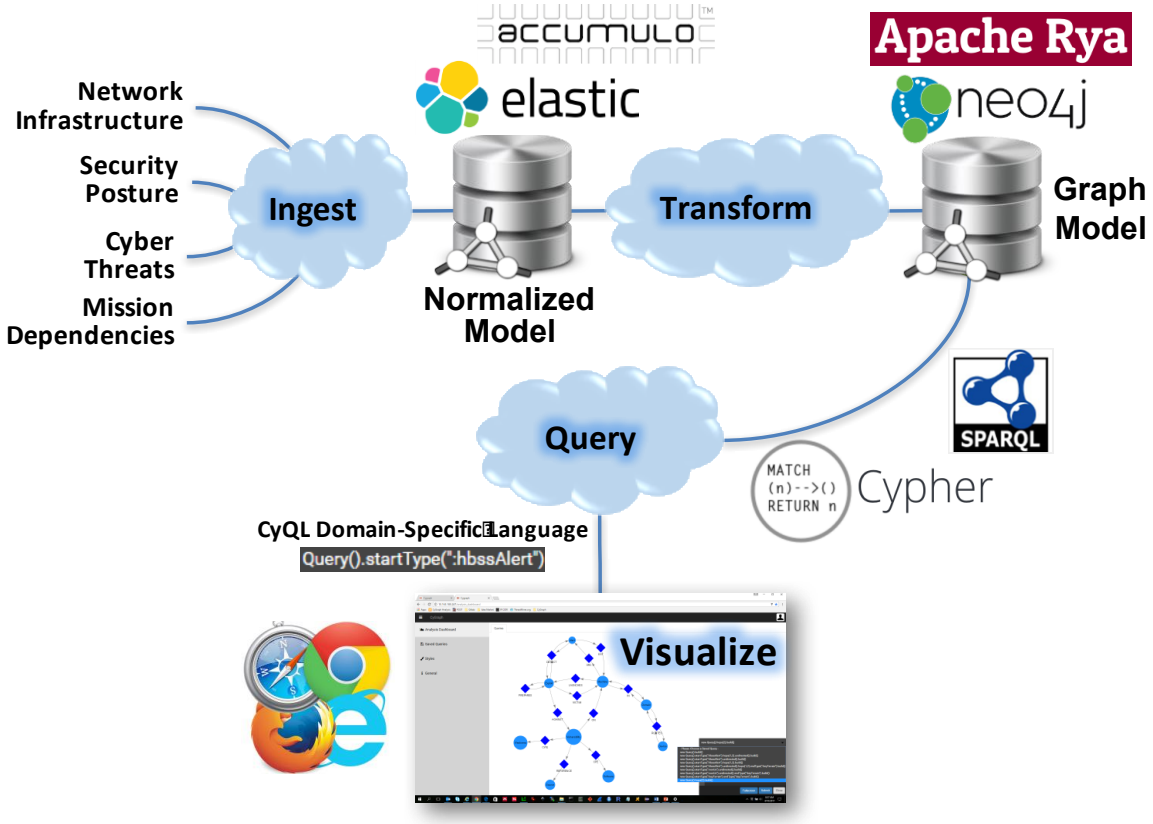

Figure 1: CyGraph Architecture.

Each of these options are available as open-source software, and (with the exception of Rya) have commercial support available. The second option (Rya+Accumulo) is available as part of DISA's Big Data Platform (BDP) [26], and offers horizontal scalability. Neo4j scales horizontally for reads and vertically for writes [27].

In the CyGraph front-end analyst dashboard, graph pattern-matching queries are expressed domain-specific query language, which CyGraph compiles to Cypher (for Neo4j) or SPARQL (for Rya). This presents a simplifying layer of abstraction, designed specifically the desired risk analysis, freeing the analyst from learning a complex general-purpose query language.

As shown in Figure 2, typical inputs to CyGraph fall under four categories:

1. *Network Infrastructure*: This captures the configuration and policy aspects of the network environment.
2. *Security Posture*: Specification of network infrastructure is combined with vulnerability data to map potential attack paths through the network.
3. *Cyber Threats*: This captures events and indicators of actual cyberattacks, which are correlated with security posture to provide context for risk analysis and attack response.
4. *Mission Dependencies*: This captures how elements of enterprise missions depend on cyber assets. This enables the relationship between system or network risks and mission risks to be expressed.

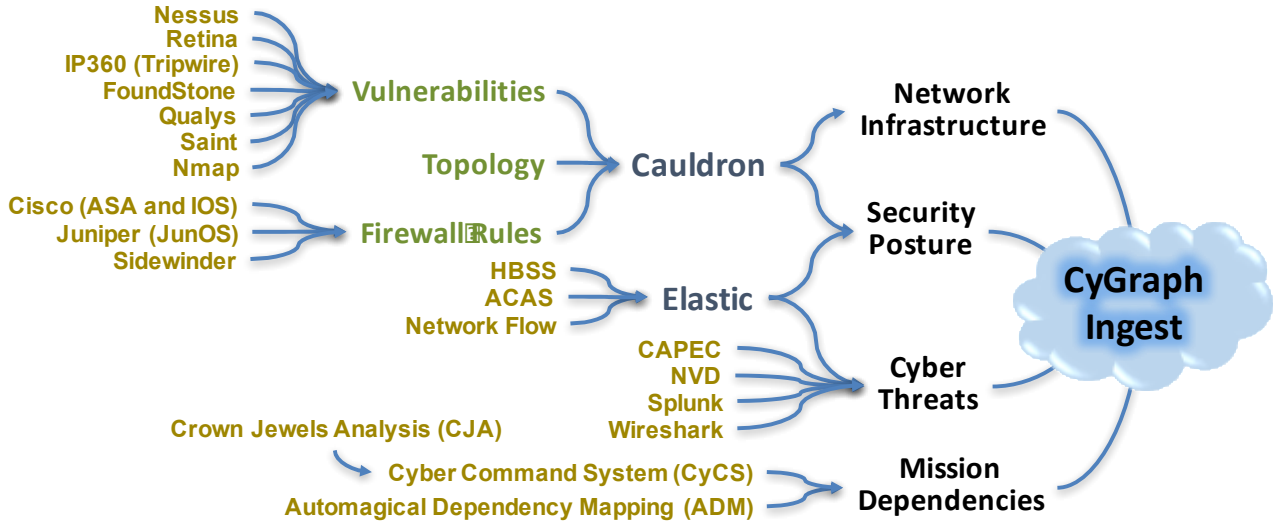

Figure 2: Example Data Sources.

The primary role of CyGraph is a knowledge base and analysis tool. It relies on other tools and data sources for raw material to build its cybersecurity graphs. For example, the Cauldron tool for Topological Vulnerability Analysis [28] [29] [30] builds network attack graphs (security posture) from host vulnerabilities, firewall rules, and network topology, which are ingested into CyGraph. For cyber threats, CyGraph ingests data for both potential and actual threats, including from the Splunk log analysis tool [31], packet capture via Wireshark [32], the National Vulnerability Database (NVD) [33], and Common Attack Pattern Enumeration and Classification (CAPEC™) [34]. For capturing mission dependencies on cyber assets, CyGraph ingests models developed through other MITRE tools [35], including Crown Jewels Analysis (CJA) [36], Cyber Command System (CyCS) [37], and Automagical Dependency Mapping (ADM) [38].

The CyGraph data model is schema-free, so that the model is decoupled from the storage implementation. The particular way in which the data are transformed to a property graph determines an instantiated CyGraph model. So, for example, an arbitrary subset of the data sources in Figure 2 need be populated for useful analysis – often only a single data source is ingested. User queries must match a given graph model instantiation. This is enforced through CyGraph's domain-specific query language. This means that the middle-tier query language translation needs to be informed if the data model is extended though new node types or edge types.

3.0 SYSTEM OPERATION

Figure 3 shows one of the panels of the CyGraph web browser user interface. In this panel, a graph model is presented that represents the node and edge types present in the CyGraph knowledge base. The user can interact with this graph model to generate queries in the domain-specific CyQL query language. Depending on what information is captured in the knowledge base, it can be characterized as a *cybersecurity knowledge base* (capturing information about security posture, particularly about vulnerabilities) or a *resilience knowledge base* (capturing both cybersecurity knowledge and information about mission dependencies on cyber assets).

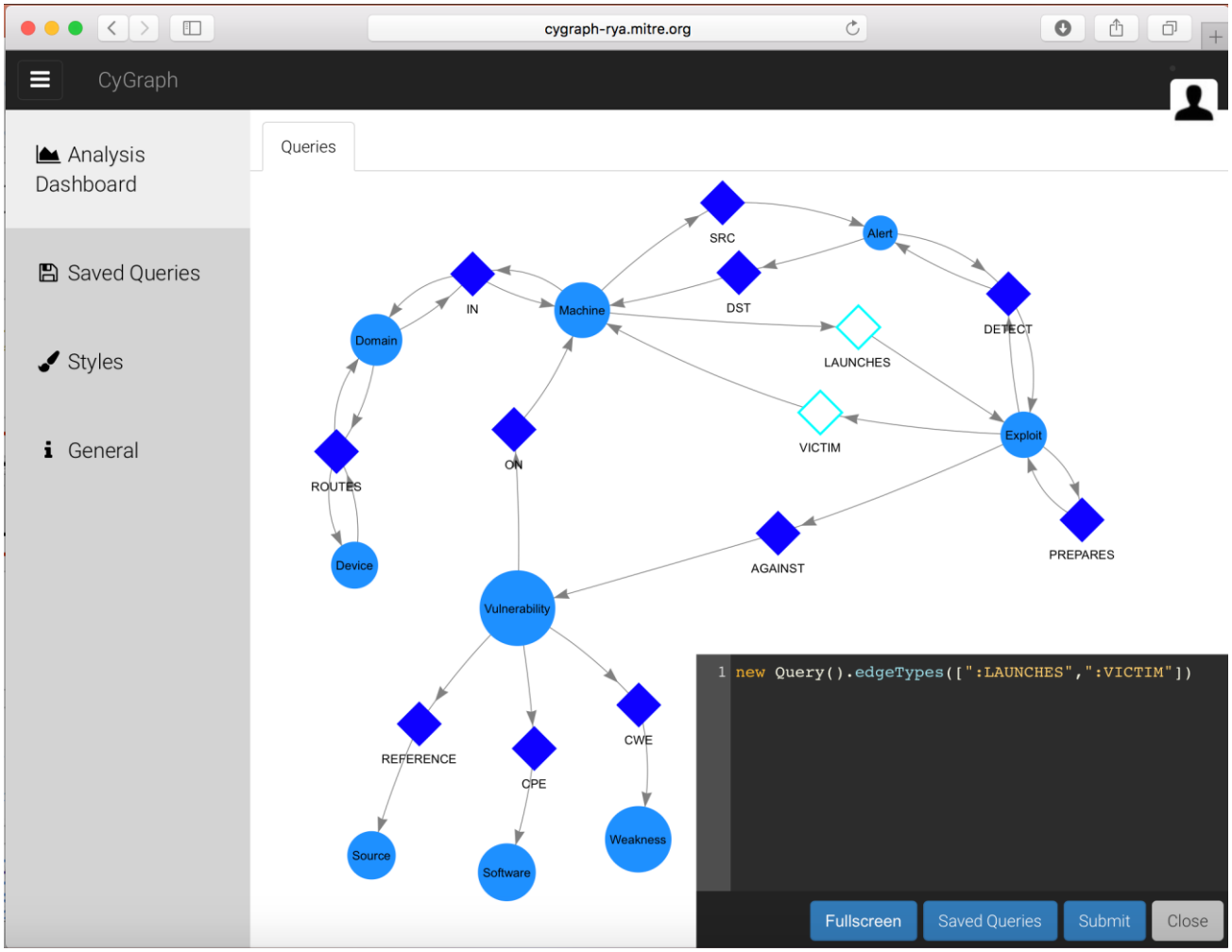

Figure 3: CyGraph Analysis Dashboard.

User-selected combinations of edge types (diamonds) populate the CyQL `edgeTypes($types)` clause, which specifies edge types to be matched in a query. For example, edges of type `IN` define relationships between `Machine` nodes and `Domain` nodes, i.e., network machine membership in protection domains (e.g., subnets) [20]. In Figure 4, Query 1 selects the `IN` and `ROUTES` edge types, which populates the `edgeTypes(":IN", ":ROUTES")` clause. This query returns the relationships among machines, protection domains, and traffic filtering devices (e.g., firewalls). Query 2 selects the `LAUNCHES` and `VICTIM` types (`edgeTypes(":LAUNCHES", ":VICTIM")` clause), matches graph elements for machines launching exploits to other (victim) machines.

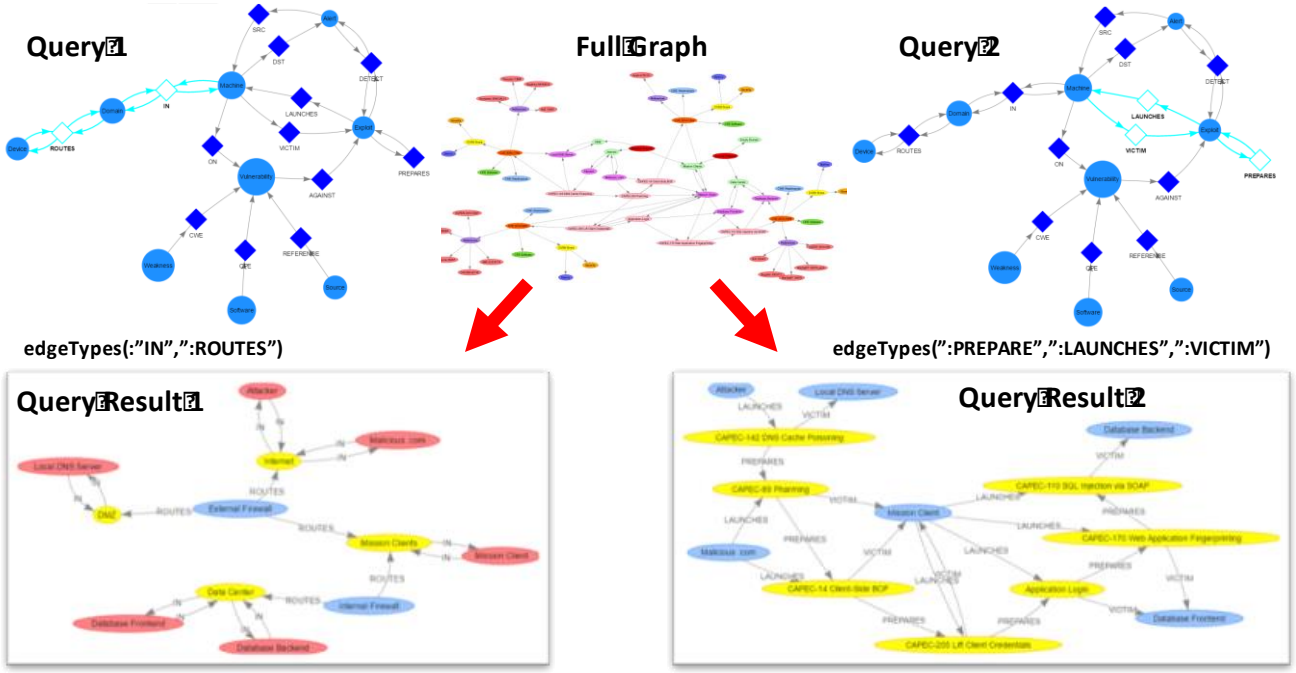

Figure 4: Selecting Edge Type Combinations in the Dashboard.

A key aspect of risk analysis is determining reachability through the knowledge base graph. Here are CyQL clauses for defining patterns of reachability:

- `hops($numHops)`: Matches multi-step graph paths of length `$numHops`. Examples include reachability from alerts (transitive risk), especially to vulnerable and/or mission-critical hosts.
- `hops($minHops, $maxHops)`: Matches multi-step graph paths of lengths between `$minHops` and `$minHops`. Provides additional flexibility for tuning reachability range, e.g., to trade off depth of search with size/complexity of query match.
- `startType($type)`: Constrains the type for starting nodes in multi-step graph paths. Typically, for selecting alert types of interest as starting points of multi-step reachability.
- `endType($type)`: Constrains the type for ending nodes in multi-step graph paths. Typically, for selecting mission-critical host types.
- `undirected()`: Controls whether graph paths are constrained to a single direction, e.g., as a pattern for lateral adversary movement.

The CyQL query language includes other features for matching patterns in the cybersecurity domain [21], including keywords for host names, IP addresses, and subnet address ranges, arbitrary Boolean combinations of clauses, and wildcards in parameter values. As shown in Figure 5, the CyGraph analysis dashboard allows queries to be stored for sharing and reuse.

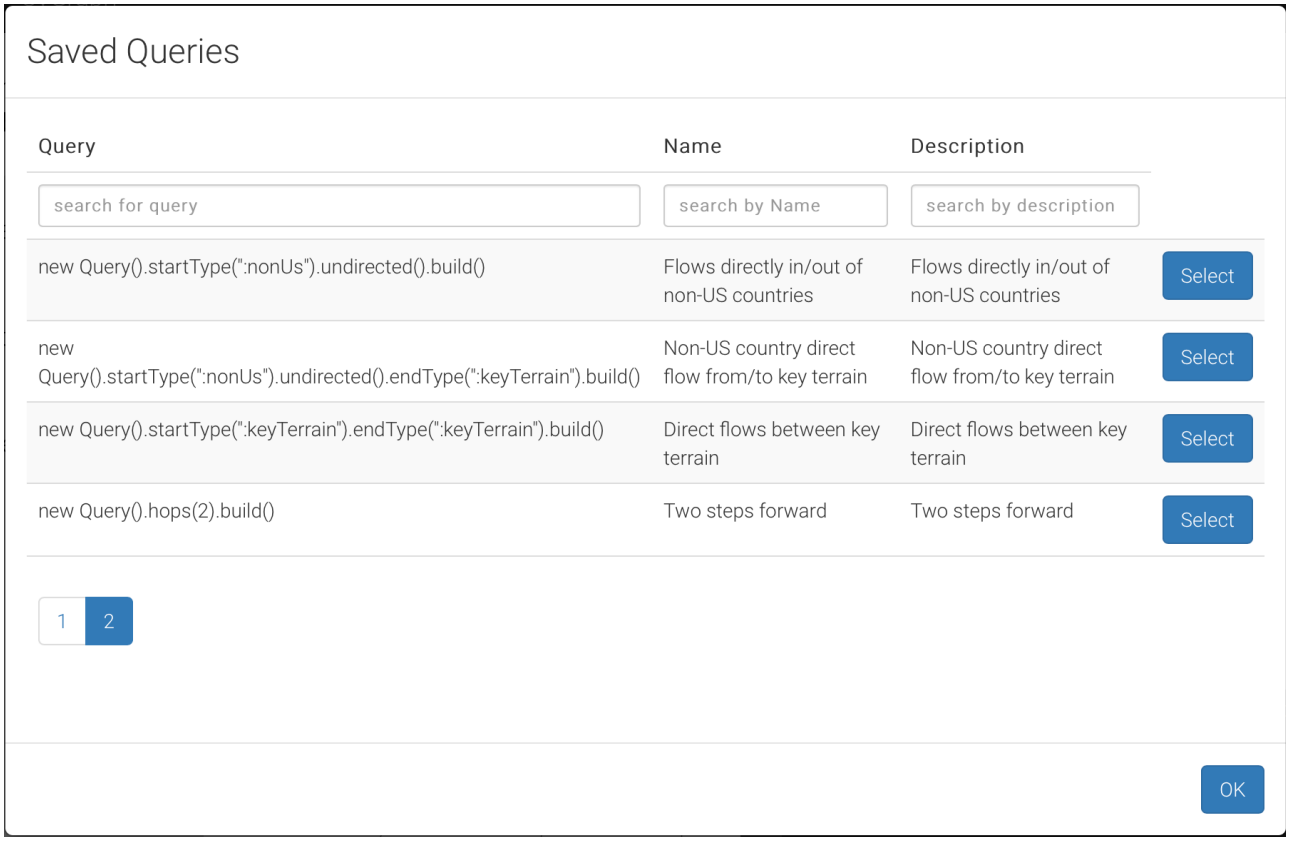

Figure 5: CyGraph Saved Queries.

4.0 RISK ANALYSIS AND RESILIENCE

We fuse information about network infrastructure, security posture, cyber threats, and mission dependencies to build a CyGraph knowledge base for enterprise risk analysis, remediation, and mitigation. There are numerous tools available for capturing information about network infrastructure, e.g., for network mapping [39], firewall rules [40], and host vulnerabilities [41]. There are also tools available that combine such information into maps of exposed vulnerabilities across networks [42] [43] [44]. Figure 6 shows such a vulnerability exposure graph generated by the Cauldron tool [42]; in previous work, such graphs are extended via CyGraph [20].

Cybersecurity knowledge graphs built from network infrastructure and security posture alone support *proactive* remediation, at least limited forms of it. For example, from the graph in Figure 6, one can prioritize vulnerabilities in terms their exposure through firewalls, and understand how adversaries can potentially exploit known vulnerability paths. It is also possible to extend this kind of analysis for measuring risk against zero-day attacks [45], as well as compute overall risk scores for enterprise networks [46] [47].

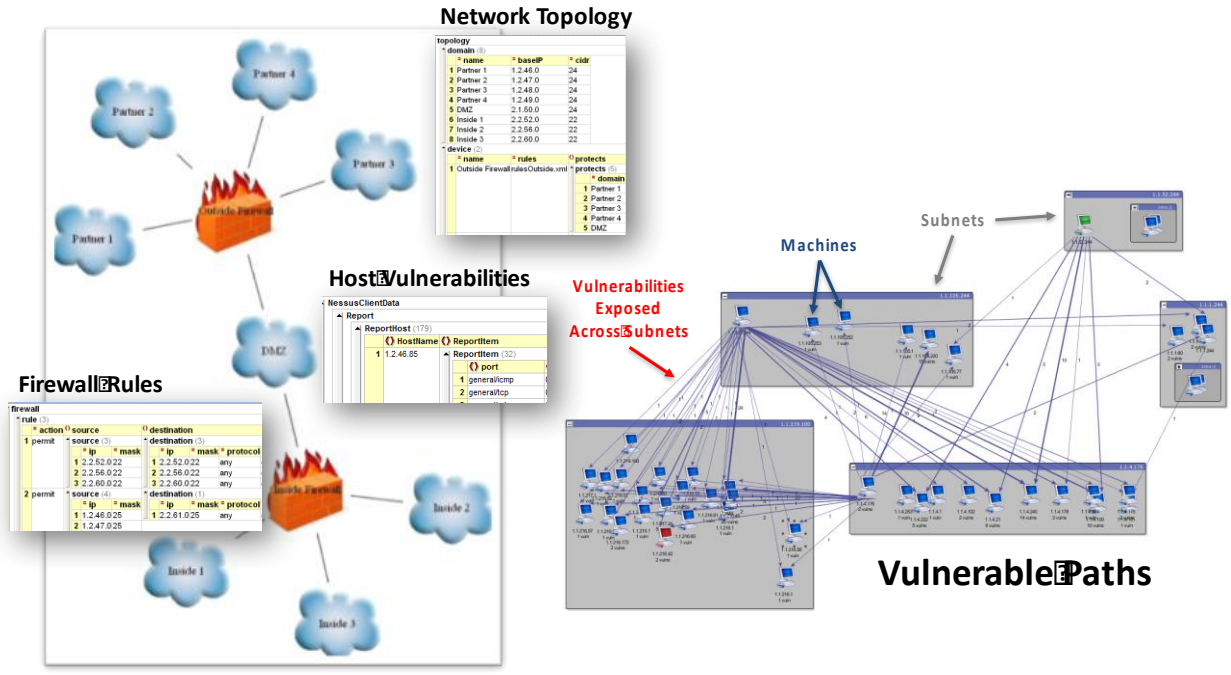

Figure 6: Mapping Vulnerable Paths through Network.

To analyse cyber resilience and enable its improvement, we extend our graph knowledge base to include how various mission functions depend on cyber assets [48] [49]. Representation of mission dependencies enables proactive remediation efforts to be mission-focused, e.g., by prioritizing vulnerable paths that lead to mission-critical cyber assets [2]. Figure 7 shows a mission-dependency graph built with the CyCS tool [37], visualized via CyGraph. This captures a hierarchy of dependencies ("needs" from higher to lower levels) among mission functions, the information needed for these functions, and the services that provide the information. Such models usually stop at the lowest level of abstract services. The actual cyber assets providing the services often change frequently, making automated dependency discovery important [38].

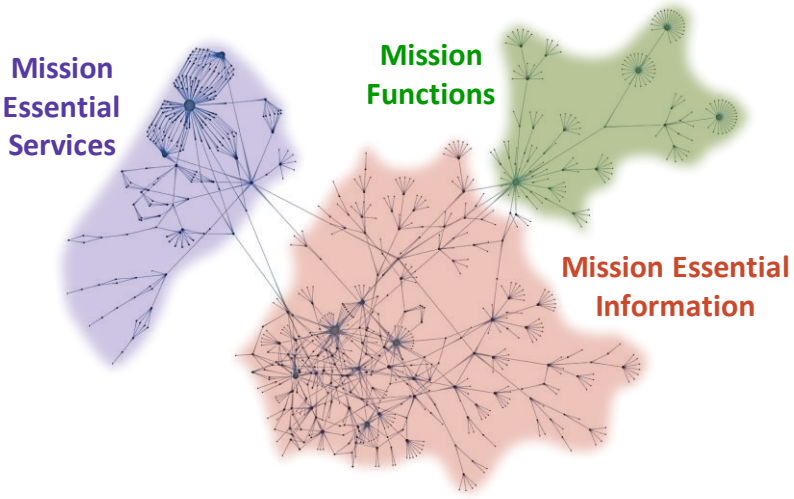

Figure 7: Dependencies among Mission Elements.

For example, by including mission dependencies in our knowledge graph, we can apply the CyQL `hops`, `startType`, and `endType` clauses to find vulnerability paths leading to mission-critical assets. We can then query relationships between vulnerability exposures and firewall locations/rules to determine exactly which rule changes on which firewalls are needed.

We further extend CyGraph knowledge bases with information on cyber threats, enabling *reactive* threat mitigation. Figure 8 shows examples of data ingested into CyGraph for such risk analysis and mitigation, i.e., from Host Based Security System (HBSS) [50], Assured Compliance Assessment Solution (ACAS) [51], and network flow records [52].

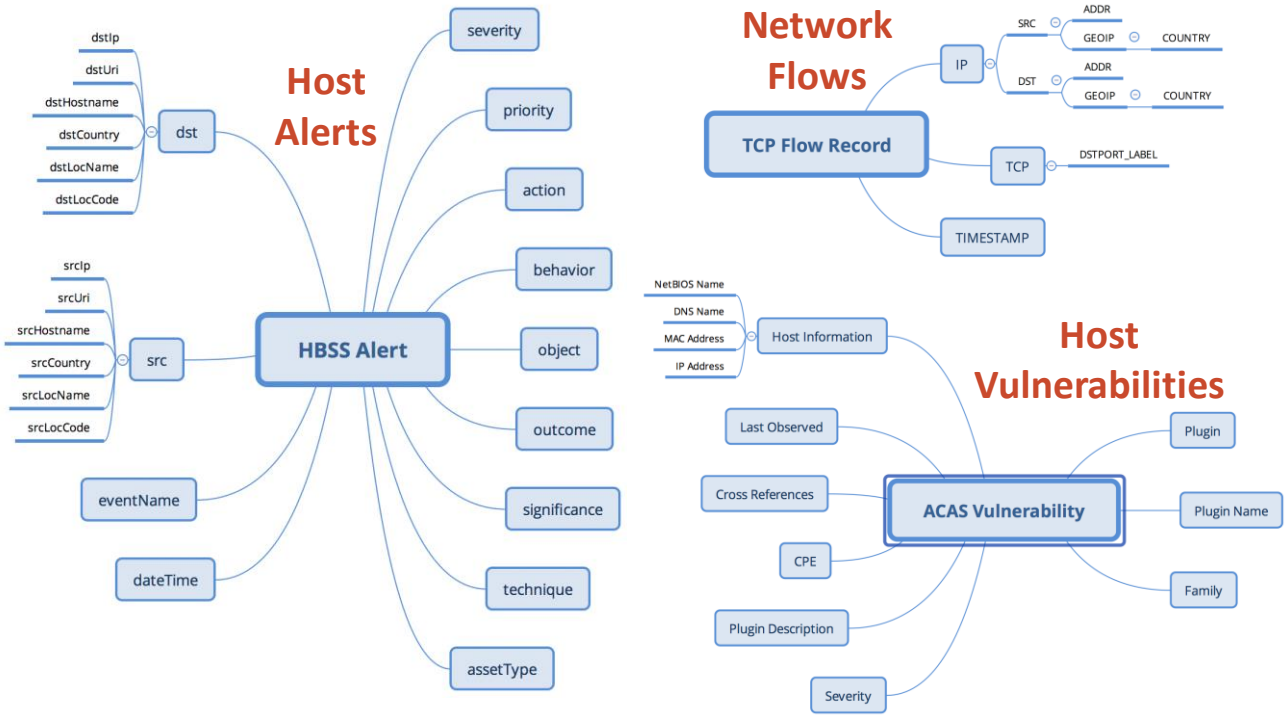

Figure 8: Alerts, Vulnerabilities, and Network Flows for CyGraph Ingest.

Figure 9 is the CyGraph property graph model based on the data sources in Figure 8. Nodes are created for each IP address, which contain relevant information from each source (sources and destinations for alerts and flows, and reported hosts for vulnerability scans). Node types are defined for categories of alerts for alert destination nodes, e.g., whether they are reconnaissance events (such a port scans), or represent actual host compromise. There are node types categorizing countries associated with network flow sources and destinations, as well as identifying key cyber assets based on services for flow destinations.

Data are continually streaming in that need to be analysed for cyber risk correlation and prioritization via CyGraph. Leveraging the open source Elastic Stack [23], the Beats platform provides agents for gathering data, with Logstash for transformation and ingest into Elasticsearch. A CyGraph web service then creates a property graph model (as in Figure 9) and imports it into the CyGraph graph data base (Neo4j). There is a similar analytic flow for CyGraph deployment on BDP, in which data streams are processed via Apache Storm [53], stored in Accumulo [25] and accessed from Rya [12].

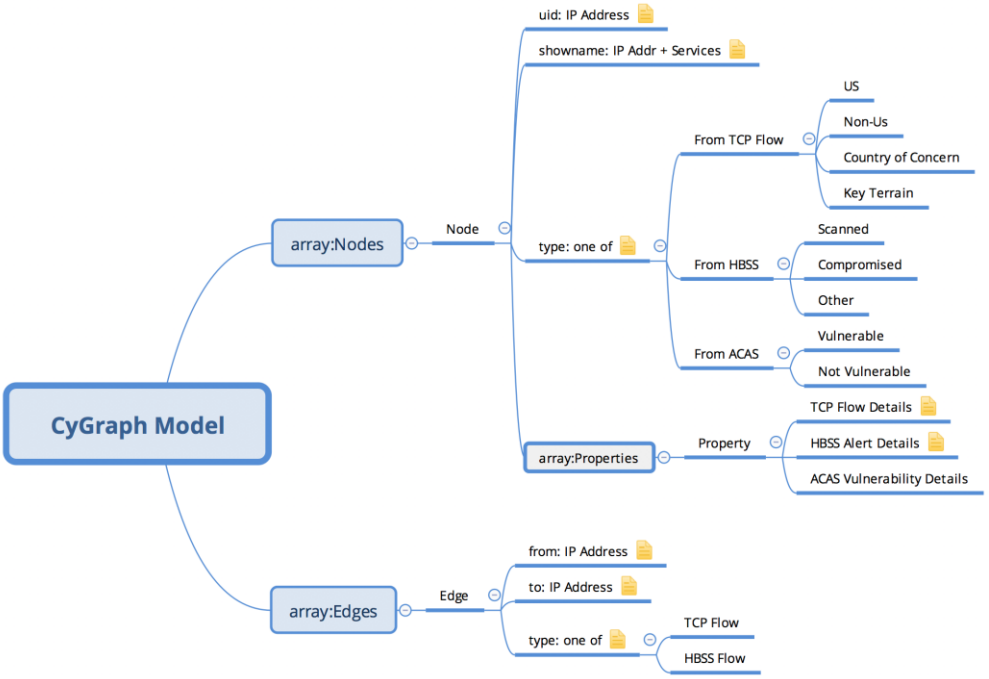

Figure 9: CyGraph Property Graph Model for HBSS, ACAS, and Network Flows.

Operationally, CyQL queries are executed on a periodic schedule from a library of standard queries, or on demand (e.g., with custom queries). In practice, usually a small fraction of the full data set under analysis matches a query. We have observed that directed paths (alerts and/or flows) are usually shallow in operational network traffic. Therefore, CyQL queries with `hops($numHops)` (which are directed) generally have few matches, especially for `$numHops` of three or more. Additional constraints, e.g., `startType($type)` and `endType($type)` tend to make query responses even smaller, while still serving to effectively identify patterns of cybersecurity risk. For example, the query `hops(2,4).startType(":compromise").endType(":keyTerrain")` matches paths of length two, three, or four, which start from a compromised node and end at a key asset node. Such paths are relatively rare but have clear implications for risk, especially when there are vulnerable nodes along a path.

The result of such a query typically looks like Figure 10, with the query match being orders of magnitude smaller than the full graph. This query match is a combination of alerts, network flows, and vulnerabilities. Here, each edge represents the set of alerts and/or network flows between a pair of IP addresses. Nodes are typed according to whether they are destinations of alerts (with the alert category), along with whether nodes are known to be vulnerable, and whether they are key cyber assets. In this fused graph representation, network flows serve to fill in potential gaps from adversary activity that was not detected by HBSS (false negatives).

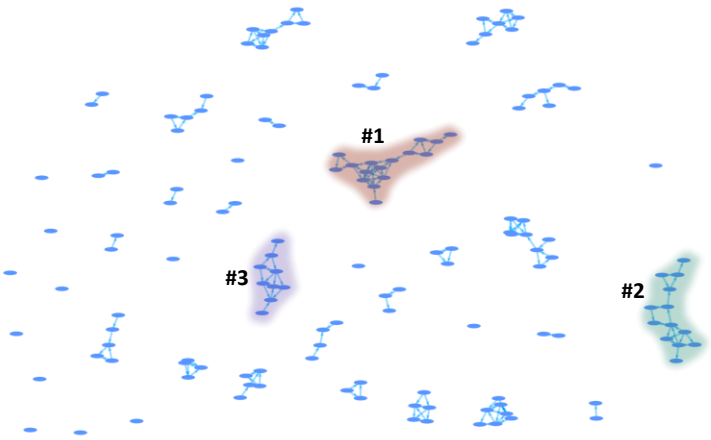

Figure 10: Prioritizing Clusters in Graph Query Results.

As shown in Figure 11, we prioritize the clusters (weakly connected components) that result from query matches. Conceptually, alerts that are not associated with other alerts (or even other traffic) are isolated events. With all other aspects of individual alerts assumed equal, those should have lower priority for defensive response. On the other hand, larger clusters of correlated alerts and other (non-alert) network flows are considered more suspicious, especially when such clusters include vulnerable hosts and/or hosts that are key assets. The choice of optimal overall risk scoring for clusters (e.g., via some utility function) remains an open problem; we apply heuristics such as the size of the cluster (connected component), with stronger weighting for numbers of key assets, alerts, and vulnerable hosts.

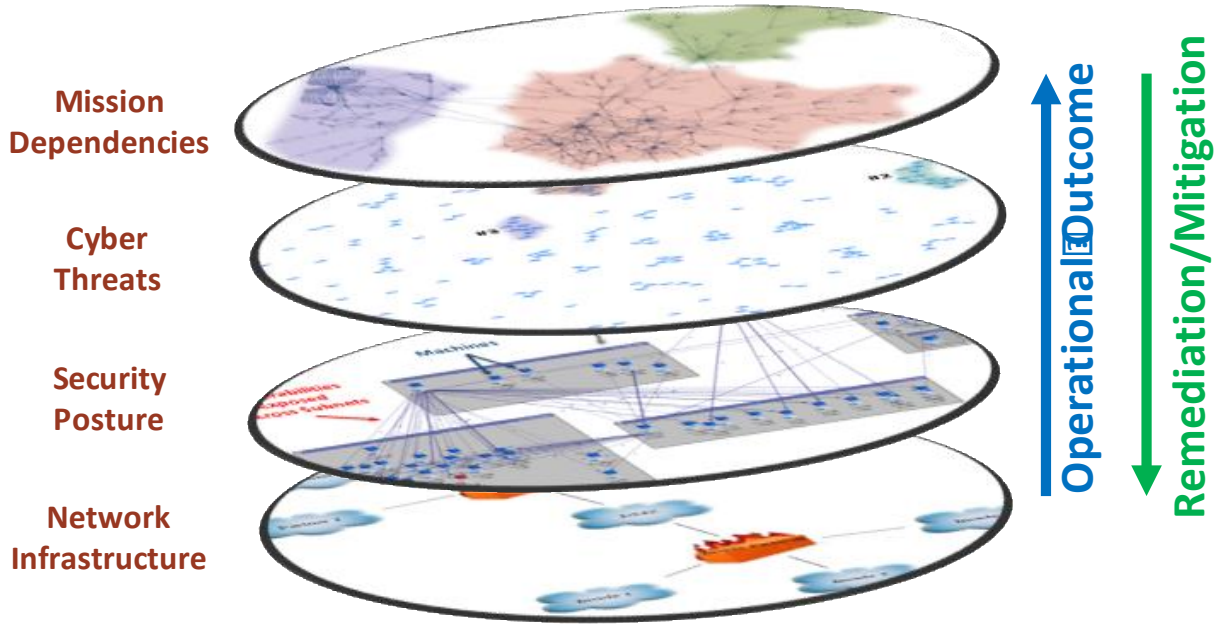

Figure 11: Graph Knowledge Layers for Cyber Resilience.

To analyse cyber resilience, relationships among aspects of network infrastructure, security posture, cyber threats, and mission dependencies must be understood and represented. By understanding these relationships, one can determine the higher priority system capabilities for applying cyber resilience. As illustrated in Figure 11, lower-level aspects tend to influence the aspects above them, in terms of maintaining mission operations in the face of threats. Security posture is influenced by elements of the network configuration (firewall rules, access control policy, web gateways, known vulnerabilities, etc.). The success of cyber threat actors is influenced by the strength of defensive posture. Mission success in turn depends on the ability of defenders to protect key cyber assets. CyGraph provides a structured yet flexible approach to incorporating these aspects into a unified knowledge base for situational awareness, risk analysis, proactive remediation, and reactive mitigation.

The application of CyGraph involves several cyber resiliency techniques, and helps achieve cyber resiliency goals and objectives. As illustrated in Figure 12, the Cyber Resiliency Engineering Framework (CREF) organizes the cyber resiliency domain into a set of goals, objectives, and techniques [1] [22]. *Goals* – Anticipate, Withstand, Recover, and Evolve – are high-level statements of intended outcomes, which help scope the cyber resiliency domain. In keeping with the fact that cyber resiliency is concerned with all threats, the goals are derived from those defined by the discipline of Resilience Engineering [54]. *Objectives* are more specific statements of intended outcomes that serve as a bridge between techniques and goals. Objectives are expressed

so as to facilitate assessment, making it straightforward to develop questions of "how well," "how quickly," or "with what degree of confidence or trust" can each objective be achieved. Cyber resiliency *techniques* characterize approaches to achieving one or more cyber resiliency objectives that can be applied to the architecture or design of mission/business functions and the cyber resources that support them. Each technique refers to a set of related approaches and technologies.

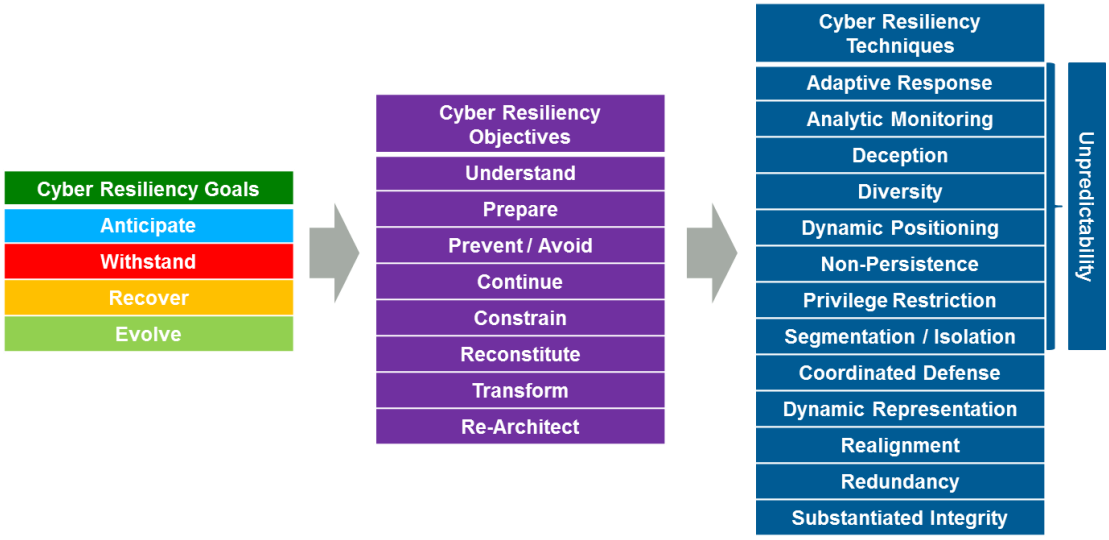

Figure 12: Graph Knowledge Layers for Cyber Resilience.

CyGraph implements the Dynamic Representation technique, which is to construct and maintain current representations of mission or business posture in light of cyber events and courses of action. CyGraph supports effective application of the other cyber resiliency techniques, enabling Adaptive Response (implement nimble cyber courses of action to manage risks), looking for gaps in Coordinated Defence (manage multiple distinct mechanisms in a non-disruptive or complementary way), and enabling evaluation of architectural alternatives in which network connectivity and device properties apply such techniques as Segmentation / Isolation, Redundancy, Diversity, Substantiated Integrity, and Non-Persistence. CyGraph can significantly improve enterprise ability to achieve the first three cyber resiliency objectives: (i) the Understand objective, to maintain useful representations of mission and business dependencies and the status of resources with respect to possible adversity; (ii) the Prepare objective, to maintain a set of realistic courses of action that address predicted or anticipated adversity; and (iii) the Prevent / Avoid objective, to preclude the successful execution of an attack or the realization of adverse conditions. Use of CyGraph to evaluate architectural alternatives also supports the Re-architect objective, to modify architectures to handle adversity more effectively.

5.0 SUMMARY AND CONCLUSIONS

Through an attributed multi-relational property graph formalism, CyGraph combines data from disparate sources, building a graph knowledge base which can be used for risk assessment and analysis of system, mission, and enterprise resilience. This knowledge base integrates information about network infrastructure, security posture, cyber threats, and mission dependencies. We use CyGraph to map vulnerability paths, mission dependencies on cyber assets, and multi-step patterns of risk among traffic flows, alerts, and vulnerabilities. Through the CyGraph resilience knowledge base, we associate risky network traffic paths with elements of access policy that allow them, for proactive remediation and reactive mitigation. Our analytic queries support use cases such as prioritizing vulnerability paths for remediation and responding to intrusion events, while focusing on the protection of key

cyber assets. Through NoSQL graph database technology with domain-specific query language, CyGraph stores and processes the resilience knowledge base at scale, while making the technology readily accessible to cyber analysts.

6.0 ACKNOWLEDGEMENTS

This work was funded by the MITRE Innovation Program (project number EPF-14-00341), with George Roelke as Cybersecurity Innovation Area Leader.

7.0 REFERENCES

[1] D. J. Bodeau and R. D. Graubart, "Cyber Resiliency Design Principles: Selective Use Throughout the Lifecycle and in Conjunction with Related Disciplines," January 2017. [Online]. Available: https://www.mitre.org/sites/default/files/publications/PR%2017-0103%20Cyber%20Resiliency%20Design%20Principles%20MTR17001.pdf.

[2] S. Musman and S. Agbolosu-Amison, "A Measurable Definition of Resiliency Using “Mission Risk” as a Metric," The MITRE Corporation, Technical Report MTR140047, 2014.

[3] The Open Web Application Security Project (OWASP), "OWASP Risk Rating Methodology," [Online]. Available: https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology. [Accessed May 29 2017].

[4] ISO/IEC, "ISO/IEC 27005:2011 Information technology — Security techniques — Information security risk management (second edition)," International Organization for Standardization, 2011.

[5] NIST, "Guide for Conducting Risk Assessments, NIST SP 800-30 Rev.1," September 2012. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-30-rev1/sp800_30_r1.pdf.

[6] M. Rodriguez, "Property Graph Algorithms," 8 February 2011. [Online]. Available: https://markorodriguez.com/2011/02/08/property-graph-algorithms/. [Accessed 31 May 2017].

[7] M. Rodriguez and J. Shinavier, "Exposing Multi-Relational Networks to Single-Relational Network Analysis Algorithms," *Journal of Informetrics*, vol. 4, no. 1, pp. 29-41, 2009.

[8] Wikipedia, "Graph database," 10 May 2017. [Online]. Available: https://en.wikipedia.org/wiki/Graph_database. [Accessed 30 May 2017].

[9] Neo Technology, "Neo4j Graph Database," [Online]. Available: https://neo4j.com. [Accessed 30 May 2017].

[10] I. Robinson, J. Webber and E. Eifrem, Graph Databases, Second ed., Sebastopol, CA: O’Reilly Media, 2015.

[11] The Linux Foundation, "JanusGraph – Distributed Graph Database," [Online]. Available: http://janusgraph.org. [Accessed 30 May 2017].

[12] R. Punnoose, A. Crainiceanu and D. Rapp, "Rya: A Scalable RDF Triple Store for the Clouds," in *1st International Workshop on Cloud Intelligence*, Istanbul, Turkey, 2012.

[13] The Apache Software Foundation, "Apache TinkerPop™," [Online]. Available: http://tinkerpop.apache.org. [Accessed 30 May 2017].

[14] P. Barcelo, "Task Force for the Design of a Query Language for Graph-Structured Data," [Online]. Available: https://databasetheory.org/node/47. [Accessed 30 May 2017].

[15] E. Eifrem, "Meet openCypher: The SQL for Graphs," [Online]. Available: https://neo4j.com/blog/open-cypher-sql-for-graphs/. [Accessed 30 May 2017].

[16] W3C Recommendation, "SPARQL 1.1 Query Language," 21 March 2013. [Online]. Available: https://www.w3.org/TR/sparql11-query/. [Accessed 30 May 2017].

[17] The Apache Software Foundation, "The Gremlin Graph Traversal Machine and Language," [Online]. Available: http://tinkerpop.apache.org/gremlin.html. [Accessed 30 May 2017].

[18] J. Chao, "Imperative vs. Declarative Query Languages: What's the Difference?," 19 September 2016. [Online]. Available: https://neo4j.com/blog/imperative-vs-declarative-query-languages/. [Accessed 30 May 2017].

[19] B. Sasaki, "Graph Databases for Beginners: Why a Database Query Language Matters," 21 August 2015. [Online]. Available: https://neo4j.com/blog/why-database-query-language-matters/. [Accessed 30 May 2017].

[20] S. Noel, E. Harley, K. H. Tam and G. Gyor, "Big-Data Architecture for Cyber Attack Graphs: Representing Security Relationships in NoSQL Graph Databases," in *IEEE Symposium on Technologies for Homeland Security (HST)*, Boston, Massachusetts, 2015.

[21] S. Noel, E. Harley, K. H. Tam, M. Limiero and M. Share, "CyGraph: Graph-Based Analytics and Visualization for Cybersecurity," in *Cognitive Computing: Theory and Applications, Handbook of Statistics 35*, Elsevier, 2016.

[22] D. Bodeau, R. Graubart, W. Heinbockel and E. Laderman, "Cyber Resiliency Engineering Aid – The Updated Cyber Resiliency Engineering Framework and Guidance on Applying Cyber Resiliency Techniques," The MITRE Corporation, Technical Report MTR140499R1, 2015.

[23] "Get Started with Elasticsearch," [Online]. Available: https://www.elastic.co. [Accessed 30 May 2017].

[24] C. Gormley and Z. Tong, Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine, Sebastopol, CA: O'Reilley Media, 2015.

[25] The Apache Software Foundation, "Apache Accumulo®," [Online]. Available: https://accumulo.apache.org. [Accessed 30 May 2017].

[26] Defense Information Systems Agency (DISA), "DISA's Big Data Platform and Analytics Capabilities," [Online]. Available: http://www.disa.mil/newsandevents/2016/Big-Data-Platform. [Accessed 30 May 2017].

[27] K. V. Gundy, "Infographic: Understanding Scalability with Neo4j," 15 July 2015. [Online]. Available: https://neo4j.com/blog/neo4j-scalability-infographic/. [Accessed 1 June 2017].

[28] S. O'Hare, S. Noel and K. Prole, "A Graph-Theoretic Visualization Approach to Network Risk Analysis," in *IEEE Workshop on Visualization for Computer Security*, Cambridge, MA, 2008.

[29] S. Jajodia, S. Noel, P. Kalapa, B. O'Berry, M. Jacobs, E. Robertson and R. Weierbach, "Network Attack Modeling, Analysis, and Response". US Patent 7,904,962, 8 March 2011.

[30] S. Noel and S. Jajodia, "Attack Graph Aggregation". US Patent 7,627,900, 1 December 2009.

[31] "What Is Splunk?," [Online]. Available: https://www.splunk.com. [Accessed 31 May 2017].

[32] "About Wireshark," [Online]. Available: https://www.wireshark.org. [Accessed 31 May 2017].

[33] "NVD – National Vulnerability Database," [Online]. Available: https://nvd.nist.gov. [Accessed 29 May 2017].

[34] "CAPEC – Common Attack Pattern Enumeration and Classification," 1 May 2017. [Online]. Available: https://capec.mitre.org. [Accessed 31 May 2017].

[35] S. Noel and W. Heinbockel, "An Overview of MITRE Cyber Situational Awareness Solutions," in *NATO Cyber Defence Situational Awareness Solutions Conference*, Bucharest, Romania, 2015.

[36] The MITRE Corporation, "Crown Jewels Analysis," [Online]. Available: http://www.mitre.org/publications/systems-engineering-guide/enterprise-engineering/systems-engineering-for-mission-assurance/crown-jewels-analysis. [Accessed 31 May 2017].

[37] The MITRE Corporation, "Cyber Command System (CyCS)," [Online]. Available: http://www.mitre.org/research/technology-transfer/technology-licensing/cyber-command-system-cycs. [Accessed 31 May 2017].

[38] S. Musman, "Automagical Cyber Dependency Mapping," The MITRE Corporation.

[39] Solarwinds, "Automatically Plot your Network in Minutes with Network Mapping Software," [Online]. Available: http://go.solarwinds.com/network-topology-mapper. [Accessed 1 June 2017].

[40] Network Perception, "Visualize and Audit your Firewalls with NP-View," [Online]. Available: http://www.network-perception.com. [Accessed 1 June 2017].

[41] Tenable, "Nessus Vulnerability Scanner," [Online]. Available: https://www.tenable.com/products/nessus-vulnerability-scanner. [Accessed 1 June 2017].

[42] CyVision Technologies, "Cyber Risk is a Growing Problem – What are You Doing to Proactively Tackle the Problem?," [Online]. Available: https://www.cyvisiontechnologies.com. [Accessed 1 June 2017].

[43] RedSeal, "Cybersecurity Analytics Platform," 1 July 2017. [Online]. Available: https://www.redseal.net.

[44] Skybox Security, "The Skybox Security Suite," [Online]. Available: https://www.skyboxsecurity.com/products/overview.

[45] L. Wang, S. Jajodia, A. Singhal, P. Cheng and S. Noel, "k-Zero Day Safety: A Network Security Metric for Measuring the Risk of Unknown Vulnerabilities," *IEEE Transactions on Dependable and Secure Computing*, 2013.

[46] S. Noel and S. Jajodia, "Metrics Suite for Network Attack Graph Analytics," in *9th Annual Cyber and Information Security Research Conference (CISRC)*, Oak Ridge National Laboratory, Tennessee, 2014.

[47] S. Noel and S. Jajodia, "Measuring Enterprise Cybersecurity Risk through Topological Vulnerability Analysis," in *Network Security Metrics and Applications*, L. Wang, Ed., Springer, 2017.

[48] S. Noel, J. Ludwig, P. Jain, D. Johnson, R. K. Thomas, J. McFarland, B. King, S. Webster and B. Tello, "Analyzing Mission Impacts of Cyber Actions (AMICA)," in *NATO IST-128 Workshop on Cyber Attack Detection, Forensics and Attribution for Assessment of Mission Impact*, Istanbul, Turkey, 2015.

[49] W. Heinbockel, S. Noel and J. Curbo, "Mission Dependency Modeling for Cyber Situational Awareness," in *NATO IST-148 Symposium on Cyber Defence Situation Awareness*, Sofia, Bulgaria, 2016.

[50] Wikipedia, "Host Based Security System," [Online]. Available: https://en.wikipedia.org/wiki/Host_Based_Security_System. [Accessed 31 May 2007].

[51] Defense Information Systems Agency, "Assured Compliance Assessment Solution (ACAS)," [Online]. Available: http://www.disa.mil/cybersecurity/network-defense/acas. [Accessed 24 May 2017].

[52] S. McGillicudy, "Flow Data is Top Source for Network Analysis," [Online]. Available: https://www.kentik.com/flow-data-is-top-source-for-network-analysis/. [Accessed 31 May 2017].

[53] The Apache Software Foundation, "Apache Storm," [Online]. Available: http://storm.apache.org. [Accessed 1 June 2017].

[54] A. M. Madni and S. Jackson, "Towards a Conceptual Framework for Resilience Engineering," *IEEE Systems Journal, Vol. 3, No. 2*, June 2009.