Smart City Cyber Security

Using Digital Twins to help define IOT security monitoring strategies

This is a proof of concept that covers the stages required to build up an ontologically defined Digital Twin, Reference Models and Instance Data within Neo4j before integration within the MS Azure Digital Twin Platform. It shows how it can support cyber security monitoring and investigation.

Smart City Cyber Security

Scenario

A near future scenario describing the monitoring of the interaction between connected vehicles such as Car, eScooters and eBikes and City Street Infrastructure for security events and cyber attacks.

The Theme for this scenario comes from a research project and presentation.

Data

Event and security event data is derived from the different assets directly or from cloud based City Infrastructure management systems and vehicle Backend systems.

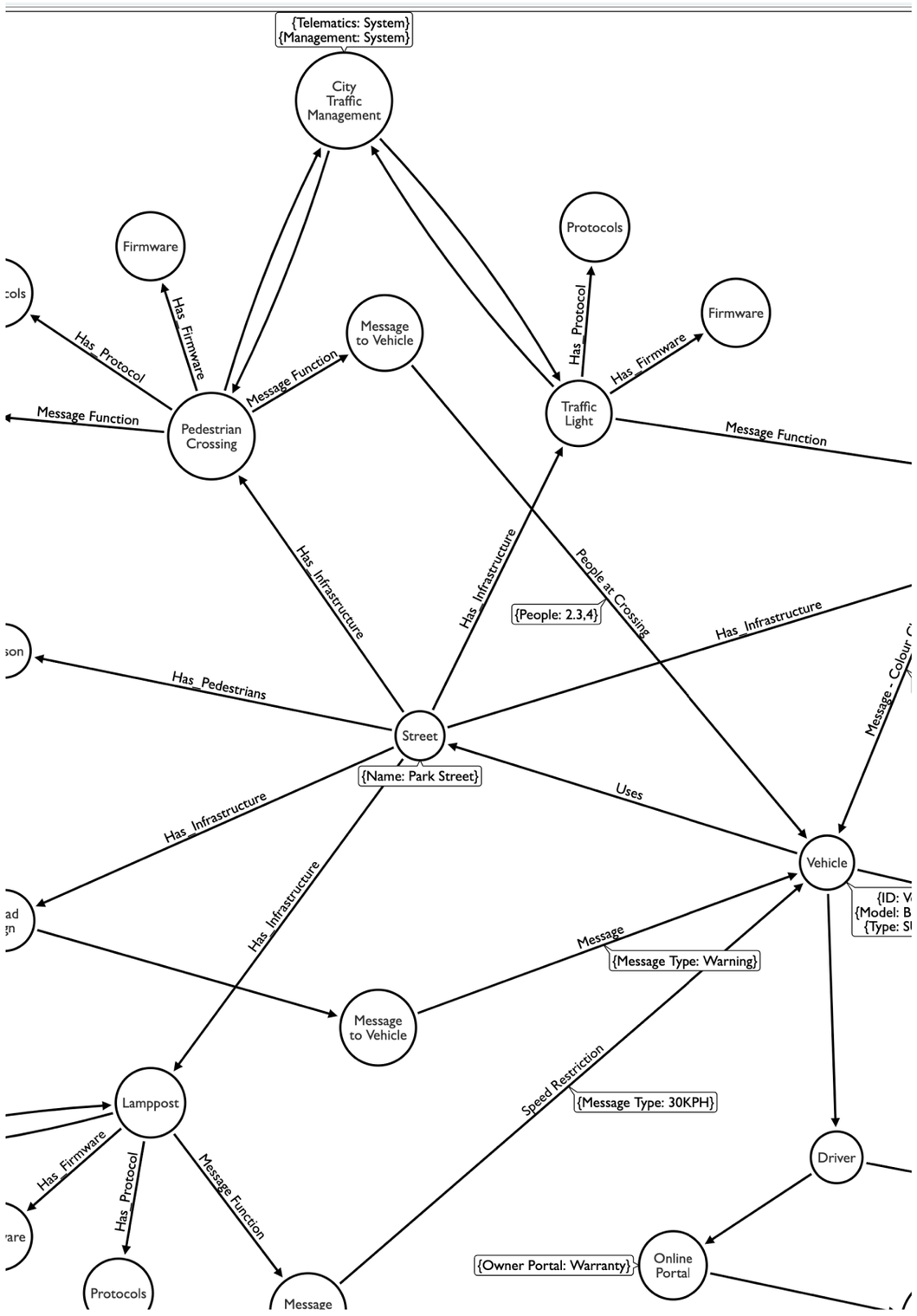
Solution

Using data models, threat models, digital twins and event data derived from asset transactions, interactions and normal operations.

The scenario also use the AWS IOT Things Graph Data Model (TDM) domain constructs to examine how data is used to support investigation and monitoring.

AWS IOT Things Graph Data Model (TDM) Security Use Cases

AWS IOT Things Graph Data Model (TDM) domain constructs	AWS Description	Security Control and Monitoring Use Case
State	The State construct is a set of properties that represent the inner state of a device at a point in time.	Determine the normal or abnormal state of operations
Mapping	The Mapping construct bridges differences across multiple representations of the same concept. It converts semantically equivalent data from one representation to another. A Mapping creates a single semantic view of data that originates from multiple sources.	Version control of configuration
Event	The Event construct describes a notification from a device that some action has been taken on it including: - The event name - A unique identifier of the source device of the event - A payload containing information that supports handling of the event	Monitor Events in relation to the result of device actions
Action	The Action construct is an abstract representation of a device performing an instance of its capability. An Action takes properties as its parameters and returns properties as its output.	Monitor Action outputs in relation to an expected outcome
Capability	The Capability construct describes a piece of functionality that is implemented by an IoT device. A Capability can extend one or more pre-existing Capabilities. It's a package containing a State and a set of Actions and Events. A Capability definition consists of the following: - A unique identifier - At most, one State - A set of Actions - A set of Events	Define the properties required to fulfil the Capability as per the expected Actions. Monitor Capability over a time period to determine normal or abnormal patterns
Device Model	The DeviceModel construct describes an abstraction of an IoT device or a stateful service. A DeviceModel must implement one Capability. It represents a conceptual device and isn't tied to any specific manufacturer. A DeviceModel consists of the following: - A TDM URN that identifies the device model - A TDM URN that identifies the device model's Capability	The parent class or master digital twin of the concept. Incorporates the Capabilities and subsequent requirements. This should construct the generalised monitoring requirements.
Device	The Device construct describes a specific IoT device that implements the Capability of a DeviceModel. This is not a concrete device, but an abstract definition of a device. After a device is defined in TDM, concrete devices can be mapped to the device definition. A Device definition includes the following: - An implementation of the parent device model's State - An implementation of the parent device model's Actions and Events in the context of a specific communication protocol, such as MQTT or Modbus	A sub-class or variant of the Device Model forming a digital twin of the specification and configuration of the Device for the purpose of association with the Asset Management of the physical device.
Service	The Service construct describes either an AWS Lambda or a RESTful web service that can be called from a Workflow. Conceptually a Service is analogous to a Device, because it can be called inside a workflow step. The structure of a Service is also similar to that of a Device. The primary difference is that the Action of a Service is a call to the web service or an invocation of a Lambda function. The Service structure contains the input and output parameters of the web service or Lambda call.	A Service associated with working with the IOT Device may exist as an End Point (Tablet or Mobile) App or Cloud application. Requires a digital twin of the service / architecture within its service boundaries and its service model. Requires a complimentary monitoring strategy.
Workflow	The Workflow construct (also called flow) describes an automated process that consists of multiple devices and stateful services. Workflow takes a set of parameters and consists of an array of steps that are connected to events. Input events can trigger a step, and a step can generate an output event. Each step can represent a Lambda function, a device action, or a web application method.	Monitoring of triggers and actions
System	The System construct describes a collection of devices, services, and workflows that interact with each other in an IoT system. For example, a security system can consist of entry sensors, cameras, light bulbs, and a door monitoring workflow. A System can be composed of other systems to create arbitrarily complex systems of systems.	Considered as part of situation awareness and broader monitoring of risk and threat intelligence.
Trigger	The Trigger construct defines the conditions that start a workflow. Triggers have two components: a condition and a set of one or more actions. The condition specifies whether to trigger a new workflow, and an action specifies what the workflow does if the condition is true.	Triggers should be classified by the severity of their actions and outcomes so critical triggers are monitored for normal or abnormal behaviours.
Deployment	The Deployment construct associates a physical location with specific devices and the triggers that start the workflows in which they are used.	Considered as part of situation awareness and broader monitoring of risk and threat intelligence.



Stage 1

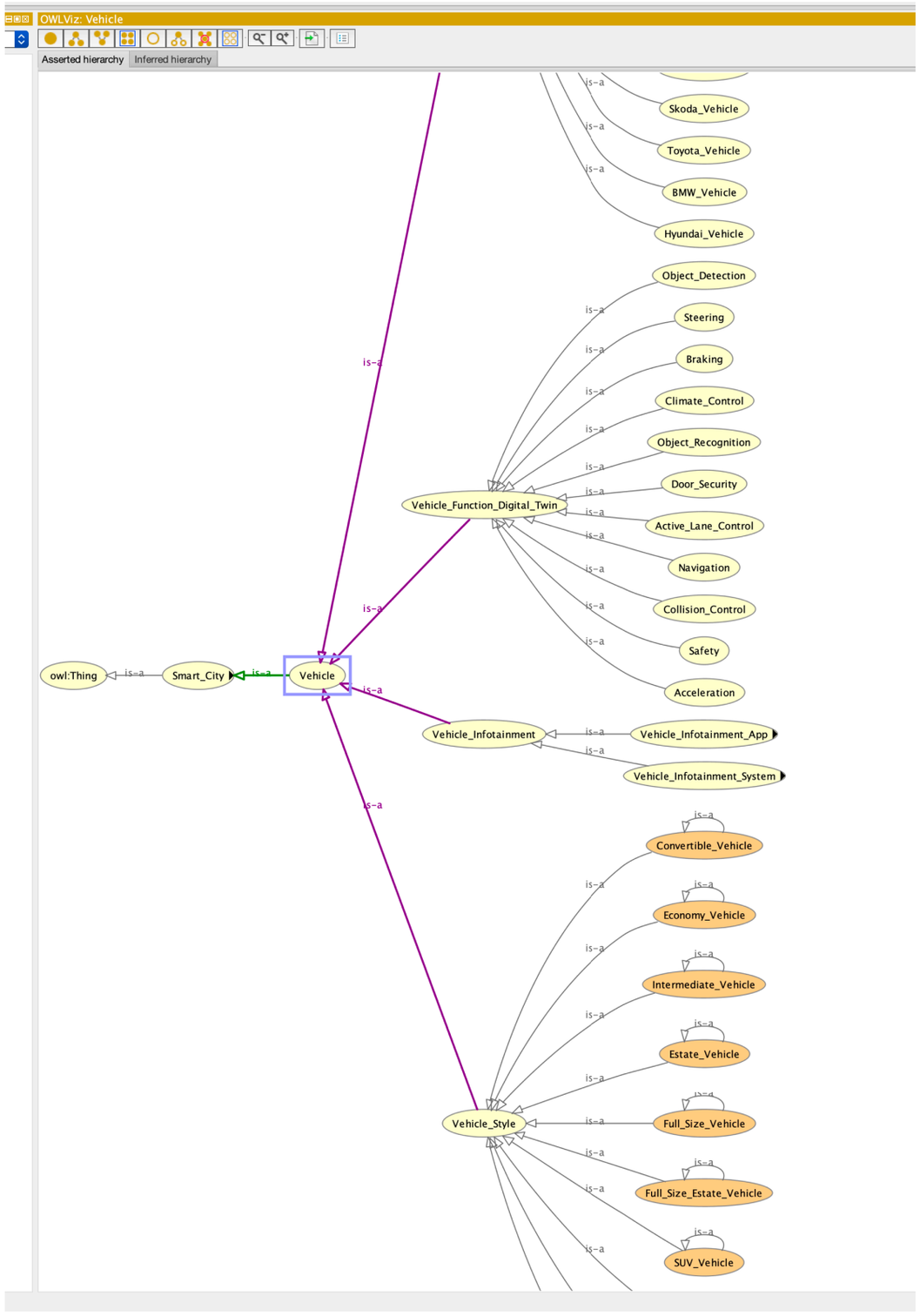
Model the Twin Environment

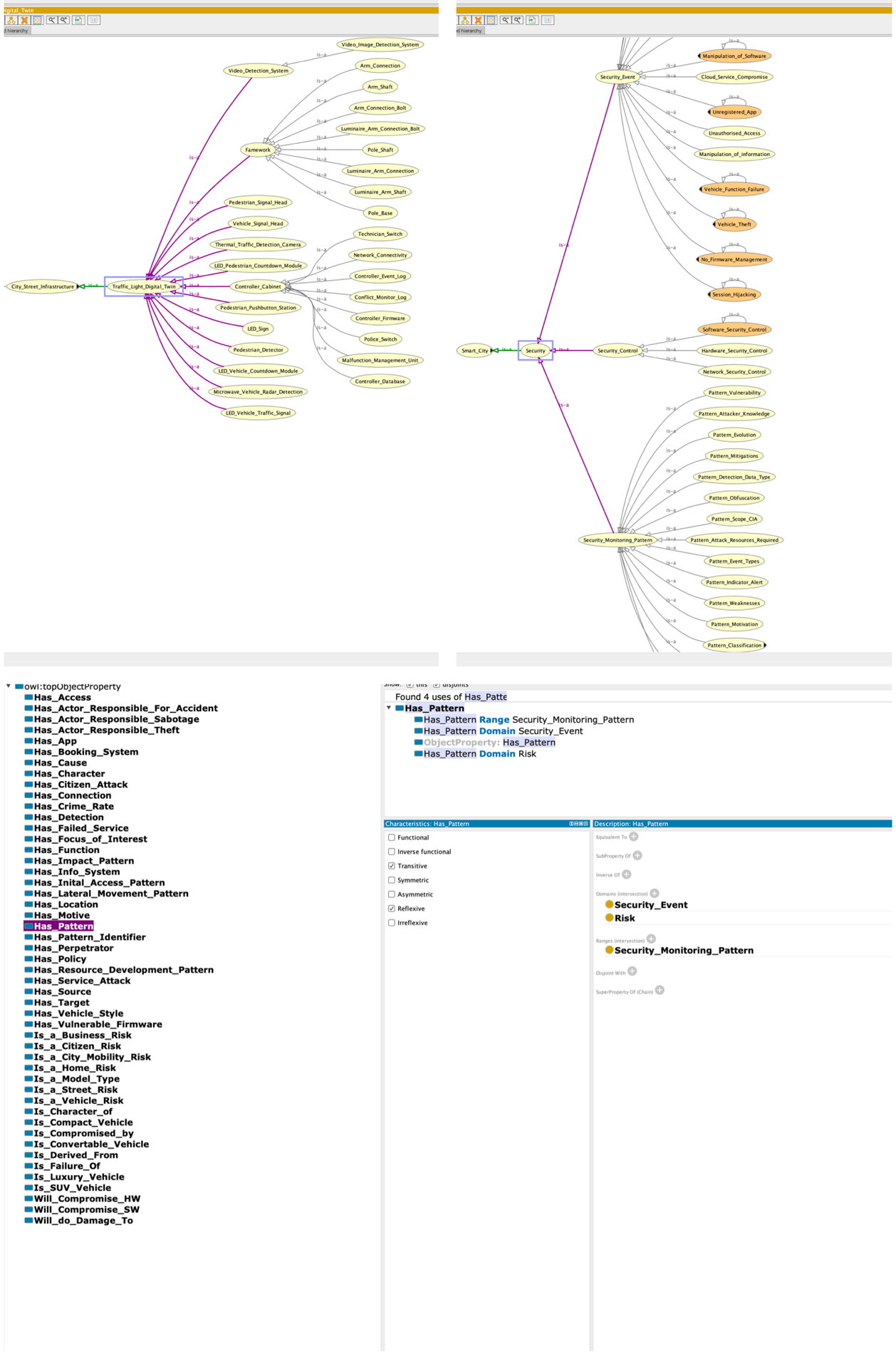
The first stage is to model the domain and Things or Devices under investigation. This example is the interaction between Vehicles and City street infrastructure such as Traffic Lights, Pedestrian Crossings, Lamp Posts and the City Traffic Management system.

Stage 2

Using an ontology development tool, such as Protege, build up models of the Things or entities in the domain model. In this example, the image shows the design of a Vehicle Function Digital Twin as a subclass of Vehicle. Other subclasses of the class Vehicle include Style, Infotainment System and Model.

The Vehicle Function Digital Twin is a collection of functions that can be subject to fail if it is impacted by misinformation from other infrastructure services such as Parking systems or Traffic Lights.



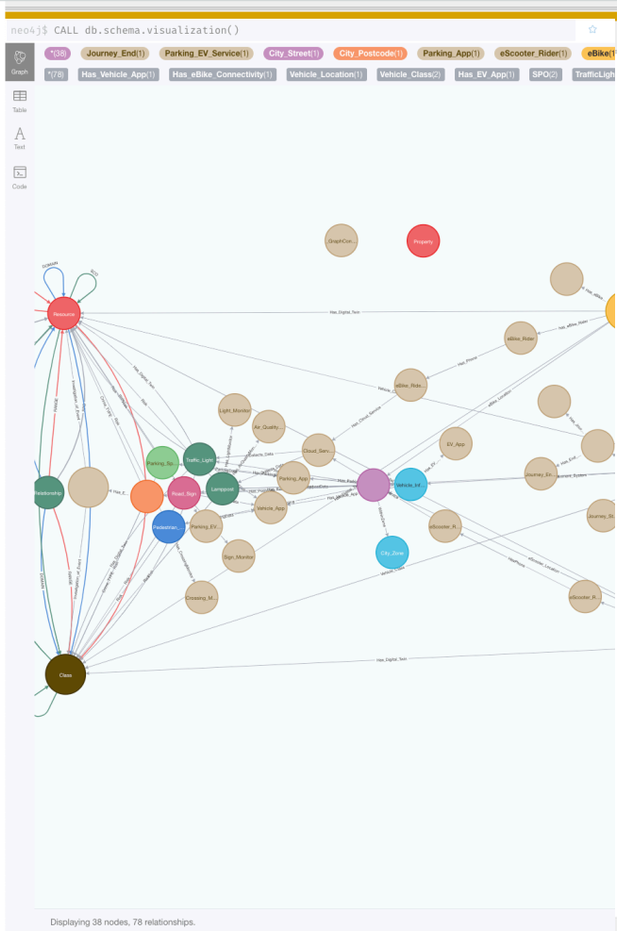


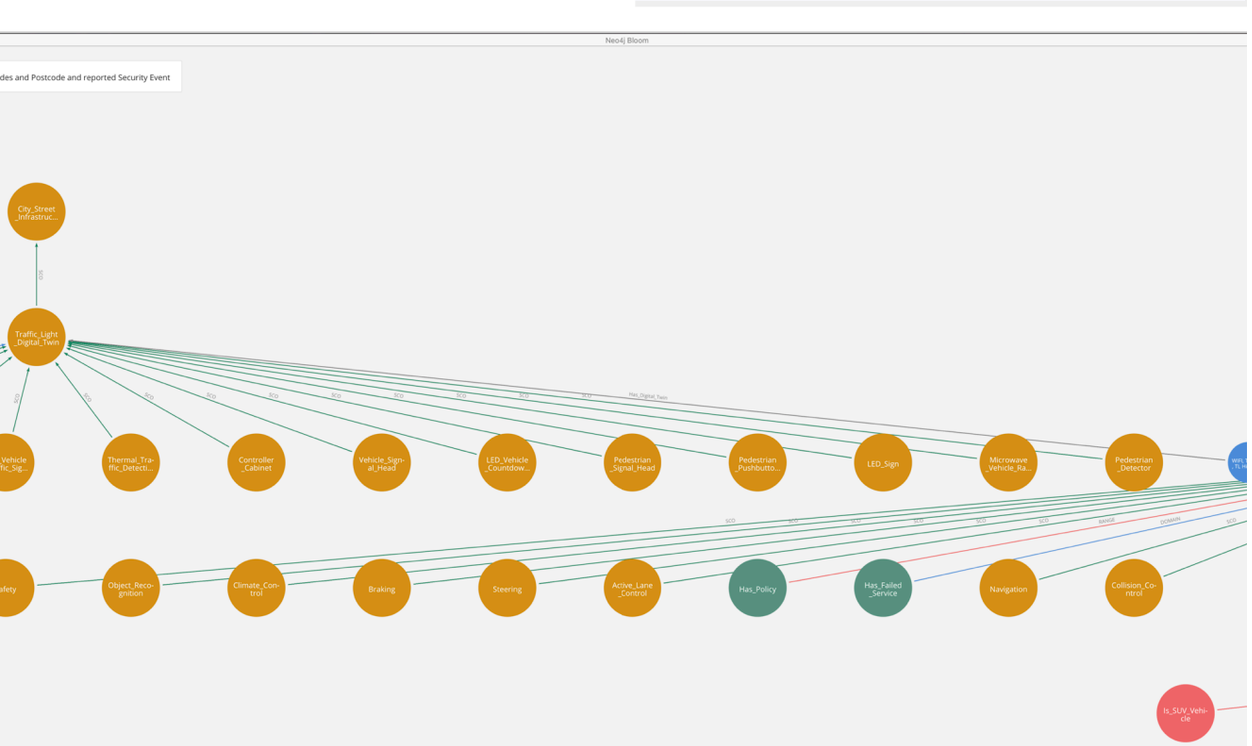
Stage 3

Continue the development of ontological models for other Things in the domain such as Traffic Lights. Extend this development with the definition and classification of the security controls, events and monitoring pattern requirements.

Each class in the ontology is supported by the definition of Class Properties, Domains and Ranges. This forms a semantic triple where the Domain Class or Subject has a Predicate or Property relationship with the Object or Range Class. This helps to enrich the data by making it more explicit and through further verification by the extended relationships.

City_Postcode	Cars_per_Postcode	Neighbourhood	City_Zone	Crime_Rate	Incident
AB1	52	Accident Blackspot	Zone 2	Crime Low	Data Manipu
AB10	8325	Accident Blackspot	Zone 4	Crime Low	Data Manipu
AB11	6488	Business	Zone 1	Crime Low	Identity Theft
AB12	15286	Downtown	Zone 7	Crime High	Ransomware
AB13	1774	Highway	Zone 1	Crime Low	Ransomware
AB14	2582	Industrial	Zone 2	Crime Low	Device Tamp
AB15	20033	Residential	Zone 3	Crime Medium	Data Manipu
AB16	10646	Retail	Zone 10	Crime Medium	Eavesdroppi
AB2	37	School	Zone 9	Crime Low	DDOS
AB21	13238	School	Zone 1	Crime Low	Data Theft
AB22	8596	School	Zone 8	Crime Low	Device Tamp
AB23	6906	School	Zone 7	Crime Low	Service Outa
AB24	9579	Suburb	Zone 3	Crime Low	DDOS
AB25	4988	Suburb	Zone 3	Crime Low	Identity Theft
AB3	6	Suburb	Zone 6	Crime Low	Vehicle Theft
AB30	4435	Suburb	Zone 3	Crime Low	Eavesdroppi
AB31	9519	Suburb	Zone 7	Crime Low	Vehicle Theft
AB32	8331	Suburb	Zone 4	Crime Low	Data Manipu
AB33	3403	Suburb	Zone 6	Crime Low	Signal Jamm
AB34	3263	Accident Blackspot	Zone 2	Crime Low	Device Hijacl
AB35	1353	Accident Blackspot	Zone 1	Crime Low	Data Manipu
AB36	338	Downtown	Zone 9	Crime High	Hardware Fa
AB37	733	Downtown	Zone 3	Crime High	Ransomware
AB38	2578	Industrial	Zone 4	Crime Low	Ransomware
AB39	9870	Industrial	Zone 8	Crime Low	Hardware Fa
AB41	12565	Public Park	Zone 8	Crime Low	Device Hijacl
AB42	17695	Public Park	Zone 10	Crime Low	Vehicle Theft
AB43	11507	Public Park	Zone 1	Crime Low	Signal Jamm
AB44	1914	Public Park	Zone 6	Crime Low	DDOS
AB45	5653	Residential	Zone 7	Crime Medium	Service Outa
AB51	20872	Residential	Zone 3	Crime Medium	Eavesdroppi
AB52	3060	Residential	Zone 6	Crime Medium	Service Outa
AB53	6941	Residential	Zone 6	Crime Medium	Signal Jamm
AB54	6052	Residential	Zone 3	Crime Medium	Eavesdroppi





Stage 4

Involves the creation of CSV data to represent the data generated by assets in the form of TDM State, Event and Action data as well as reference sources to form the Instance data set. This is imported into a Graph Database such as Neo4j.

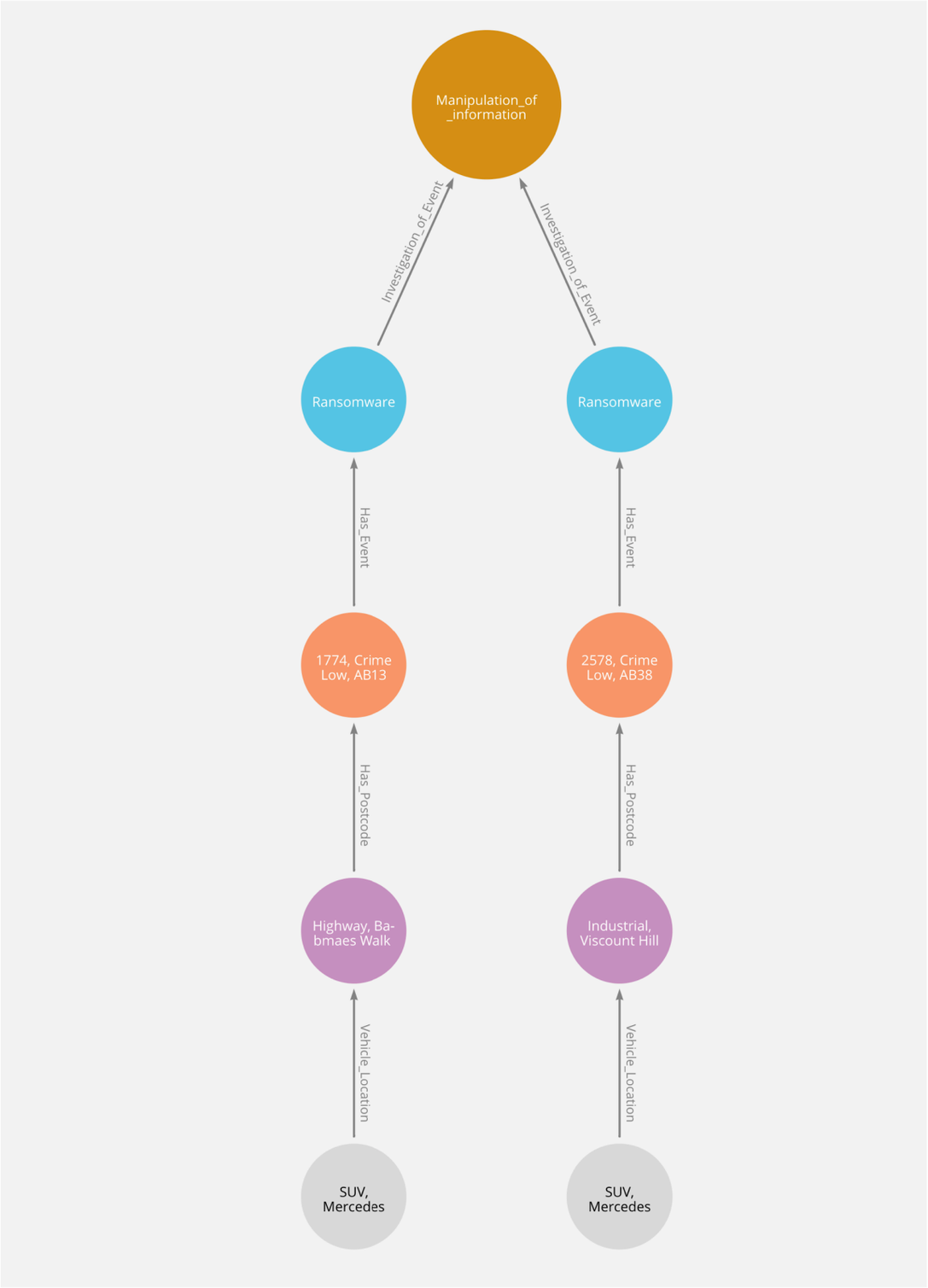
The OWL ontology is then imported into Neo4j through the use of the Neosemantics plugin.

Using Neo4j Bloom allows the Instance data to be viewed as Nodes and Relationships and extended through association or inferences to Reference Class models and Digital Twin Class models.

Stage 5

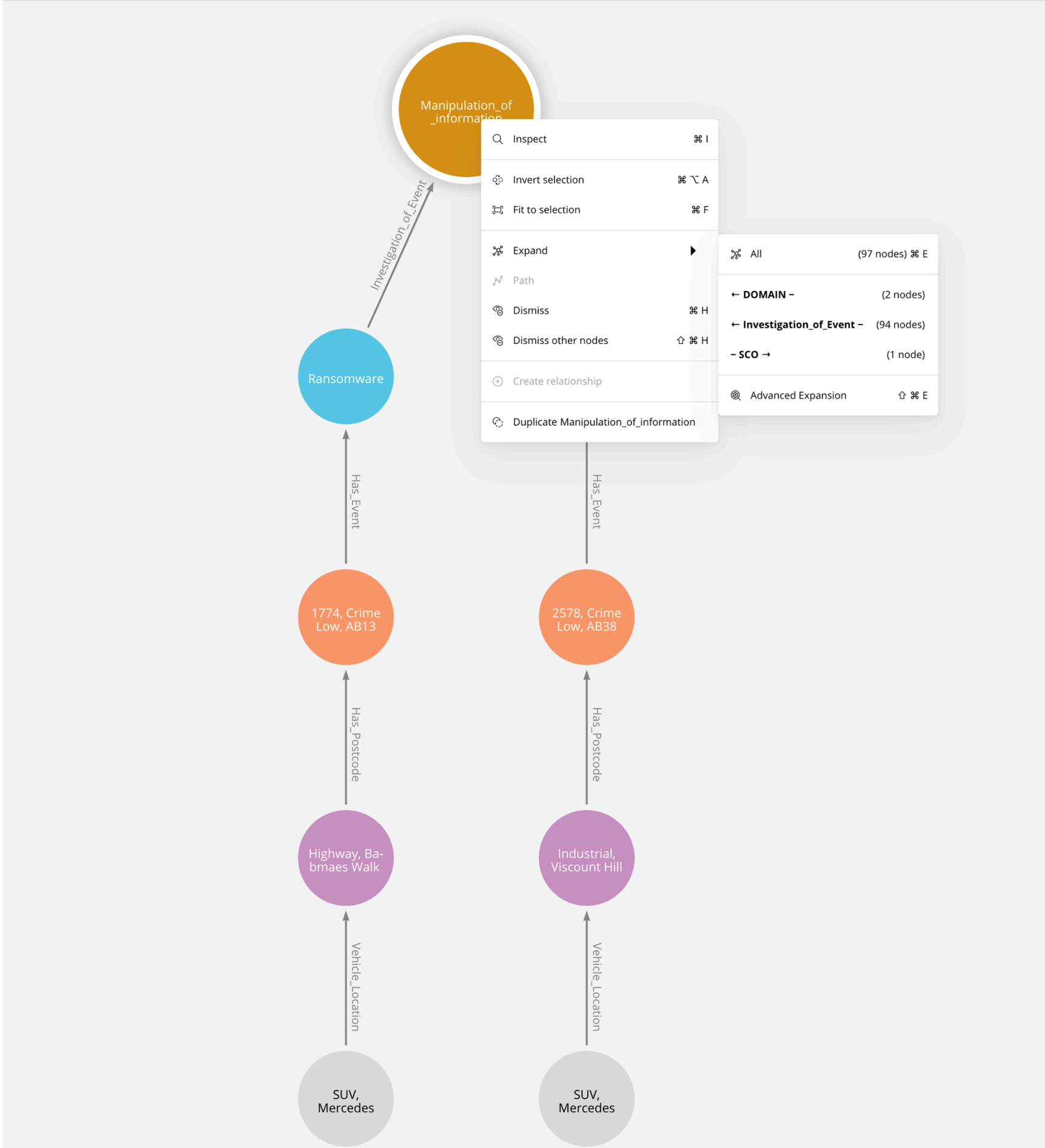
Once all the data is uploaded and processed in Neo4j the data patterns and relationships can then be explored. In this example, the larger circles represent Reference and Digital Twin classes and the smaller ones the instance data. Here we have data showing that two Vehicles at different locations have recorded a Ransomware security event.

A search query in Neo4j identifies the Nodes and Relationships and the association with the Reference Class - Manipulation of Information.



Stage 6

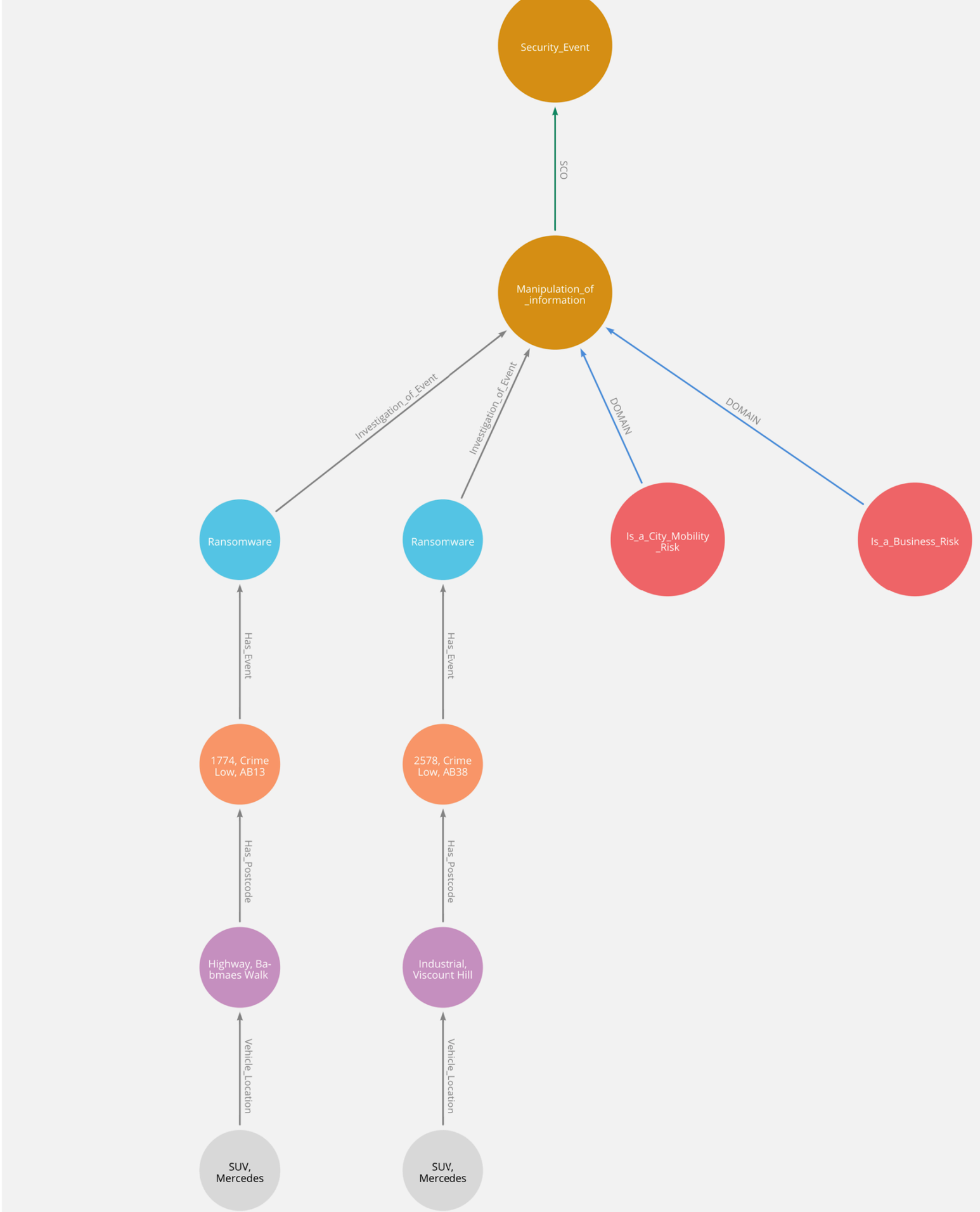
The investigation of the event and data begins by selecting and expanding the Reference Class - Manipulation of Information. This extends the Node to Domain and Subclass (SCO) references.



Stage 7

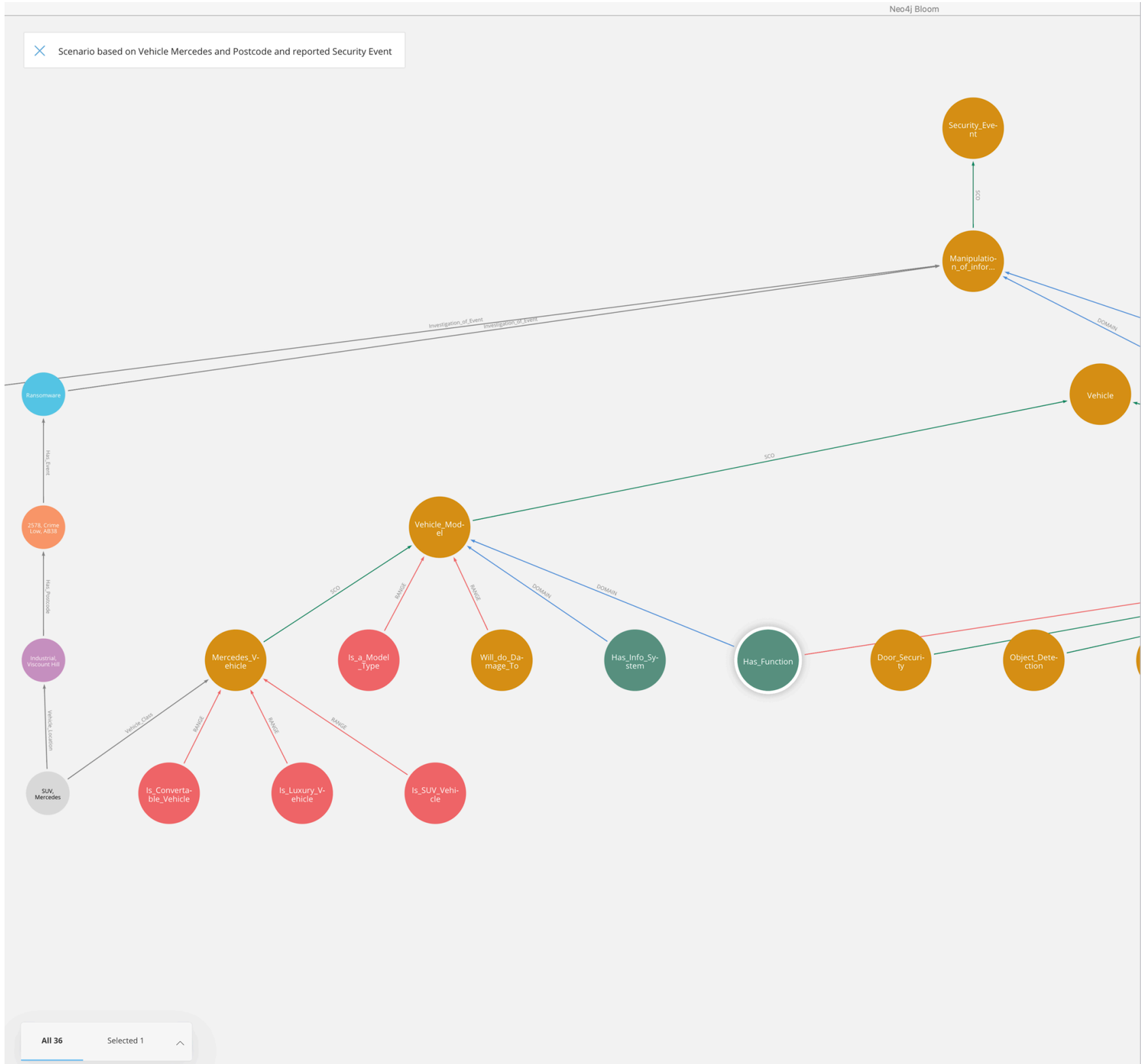
The Manipulation of Information Node is a Subclass of Security Event and is the Subject of two Predicates - Is a City Mobility Risk and Is a Business Risk. The Ranges or Object Classes would refer to different kinds of Mobility Assets such as Vehicles, eBikes or Autonomous Vehicles.

Investigation could infer that this event may impact on other Assets with similar properties or vulnerabilities or could have occurred elsewhere at other City locations.



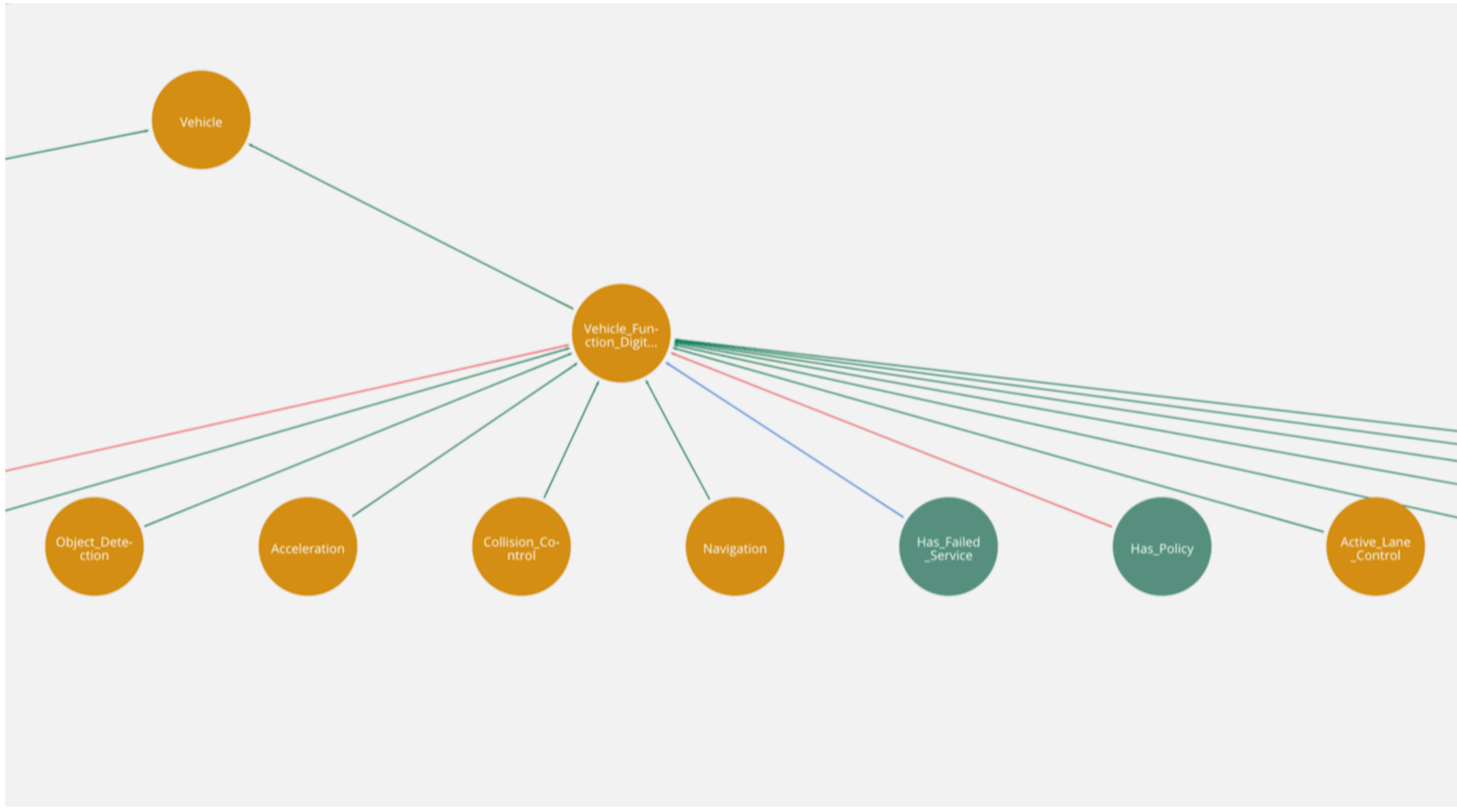
Stage 8

By extending the Vehicle Node the vehicle class relationship extends to the Vehicle Model and Vehicle parent class. This reference model provides further subclass and predicate information and a Vehicle Digital Twin model of vehicle functions.



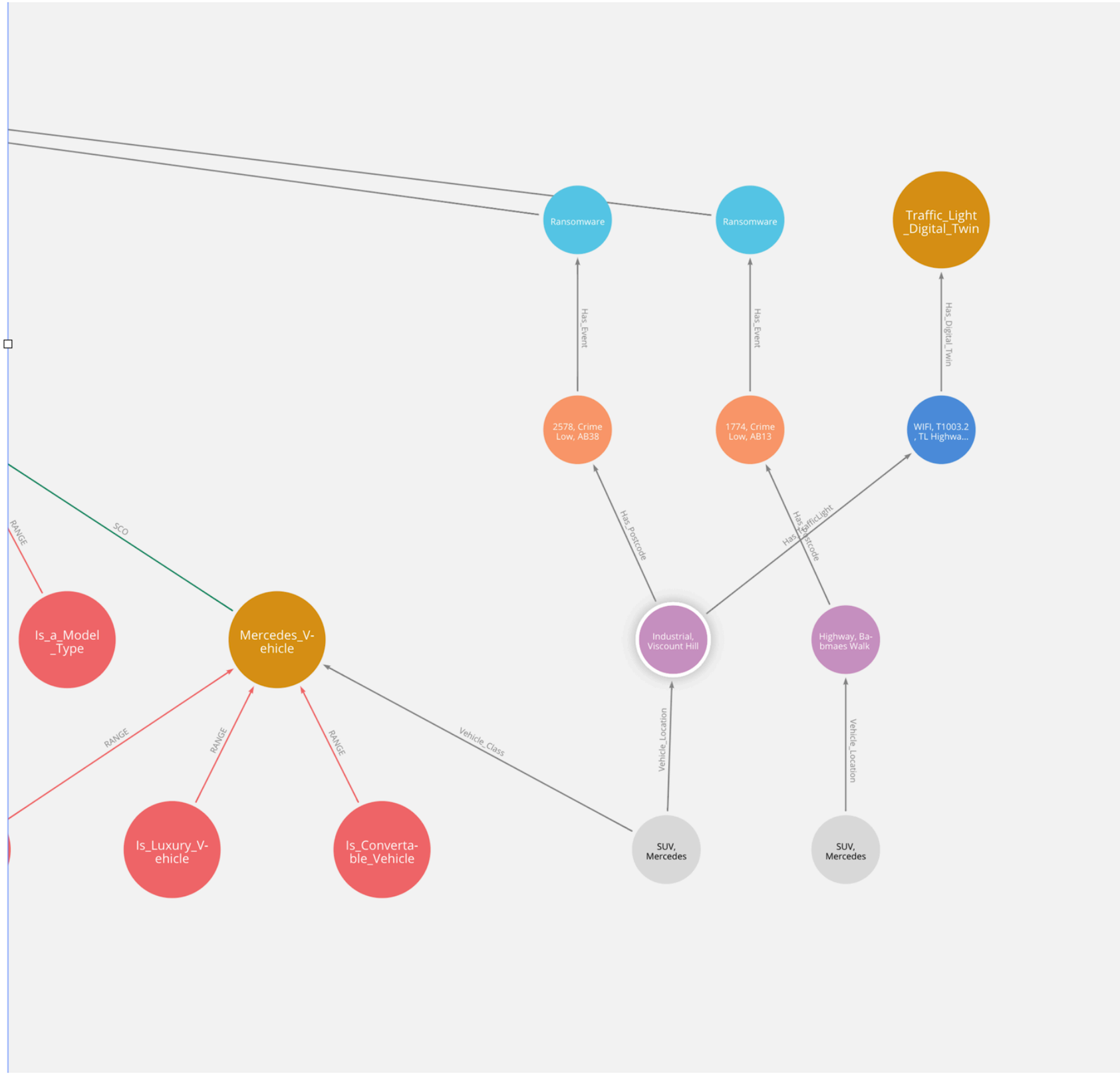
Stage 9

The Vehicle Digital Twin model breaks down into vehicle functions such as Acceleration, Object Detection and Navigation. A domain predicate - Has Failed Service - links to known vulnerability or service history and a range predicate - Has Policy - that refers to Vehicle policy controls such as Security that would detail rules or thresholds.



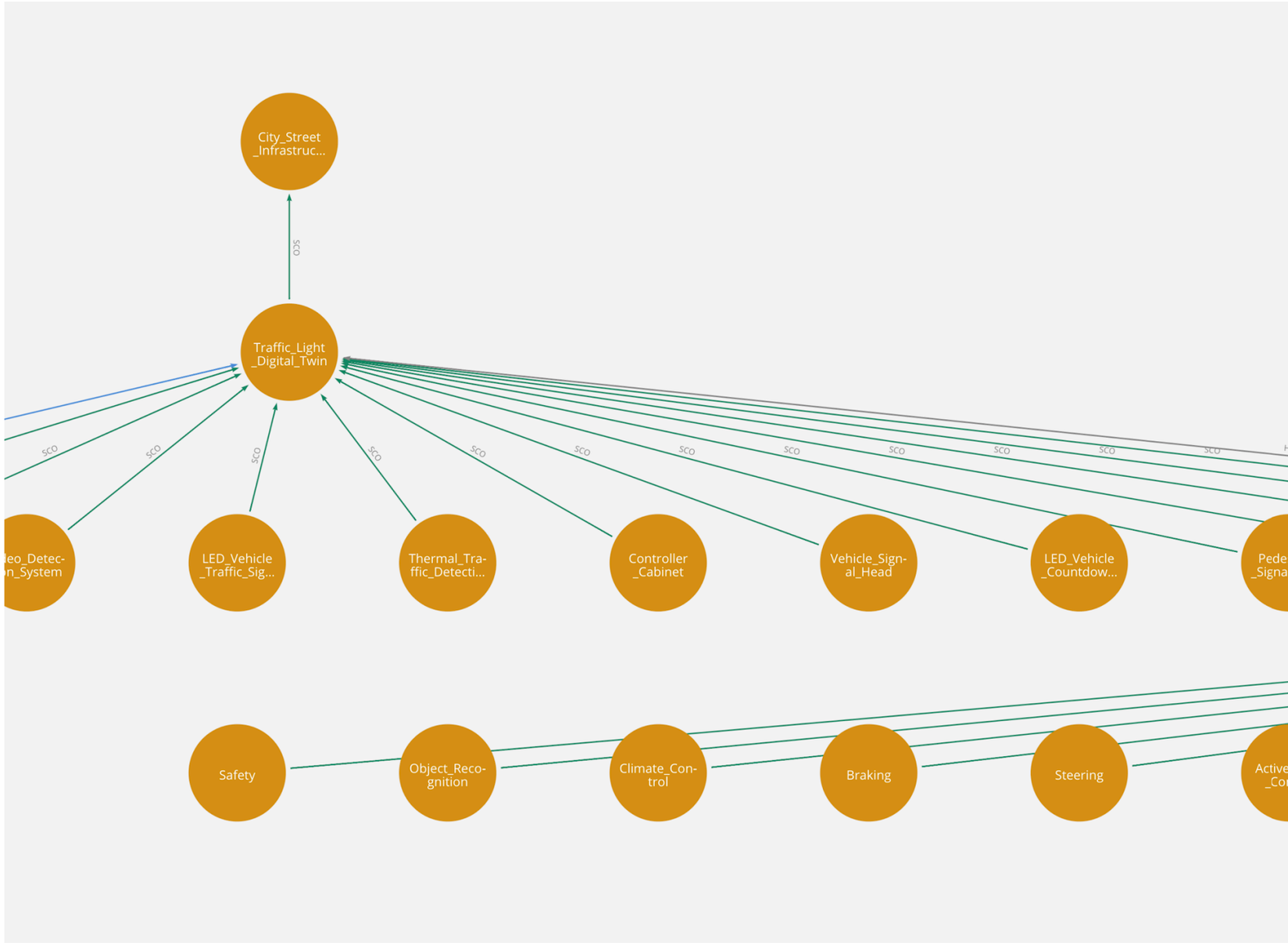
Stage 10

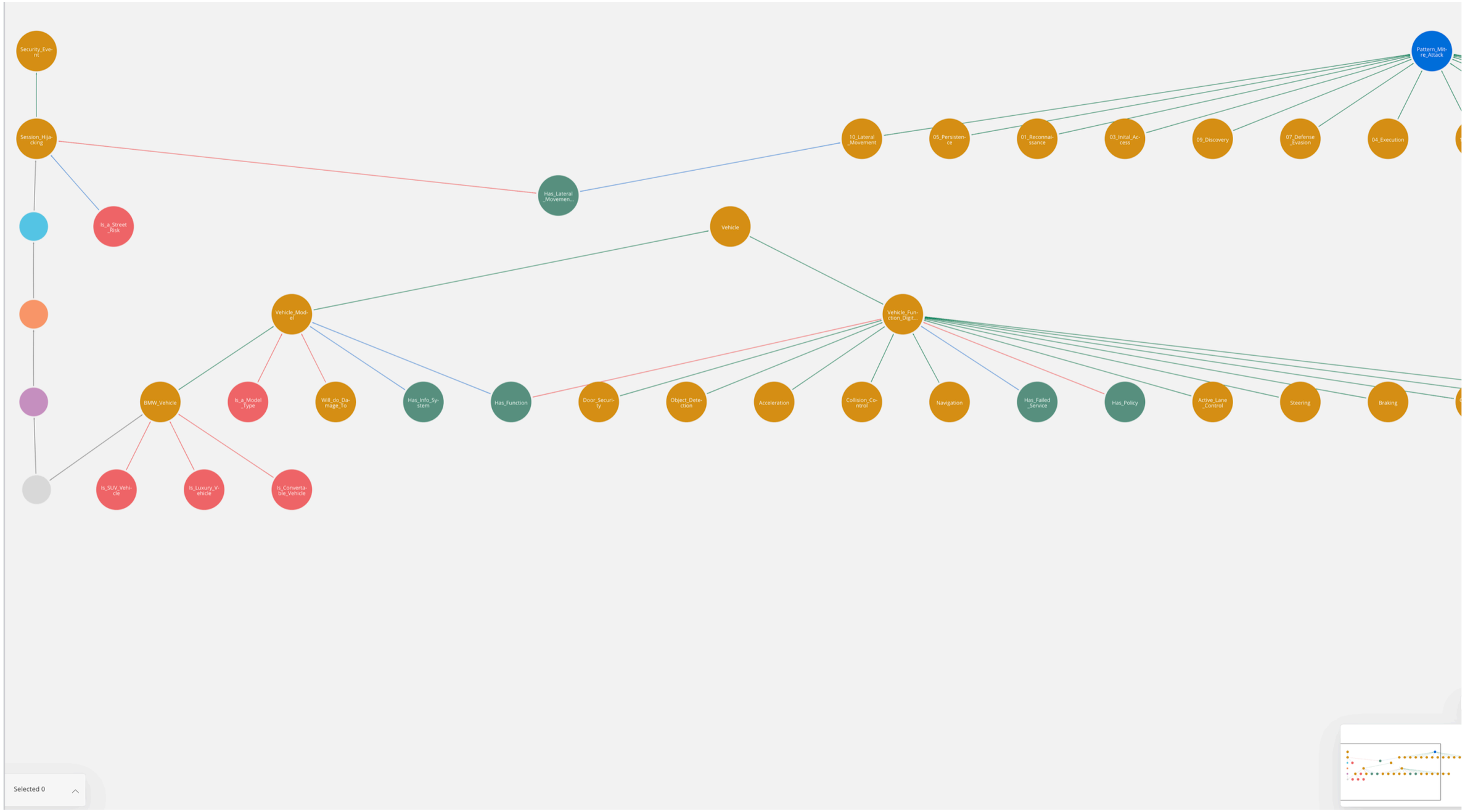
By extending the Vehicle location or Street Node to show that a Traffic Light was in proximity, the relationship to the Traffic Light Digital Twin may indicate if there were the Device Capability (TDM) to communicate with the Vehicle and if this could be the possible source of the malware.



Stage 11

The Traffic Light Digital Twin has multiple Subclasses, that show Device Capabilities (TDM), that would breakdown further into specific class types showing features and properties of the Device (TDM) that could communicate with Vehicle functions.





Stage 12

A final extension relates the security event to a Mitre Att&ck pattern class to help with evidence of known attack methods and vectors. Each extension of the original Nodes was able to provide either reference knowledge or additional support for an investigation through the examination of a technology Device Digital Twin and its parts and functions. This final relationship offers threat hunting research scenarios.

Proof of Concept

1) Building data models and Digital Twin
Protege provides the platform to design and develop ontological reference models and Digital Twins.

2) Scenario planning with the data and Digital Twins
Neo4j provides a platform to upload and analyse multiple instance data sets and with the use of the Neosemantics plugin provides:

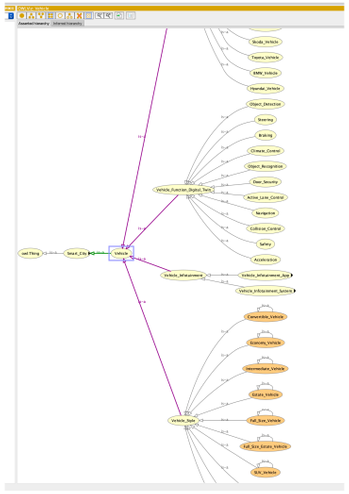
- import of RDF
- import OWL ontologies
- model mapping
- Graph model scenarios
- basic inferencing

3) Building the Digital Twins
Using MS Visual Code DTDL, MS DT Explorer and the MS Azure Digital Twin platform to define and build the Digital Twins designed in step 1 and 2

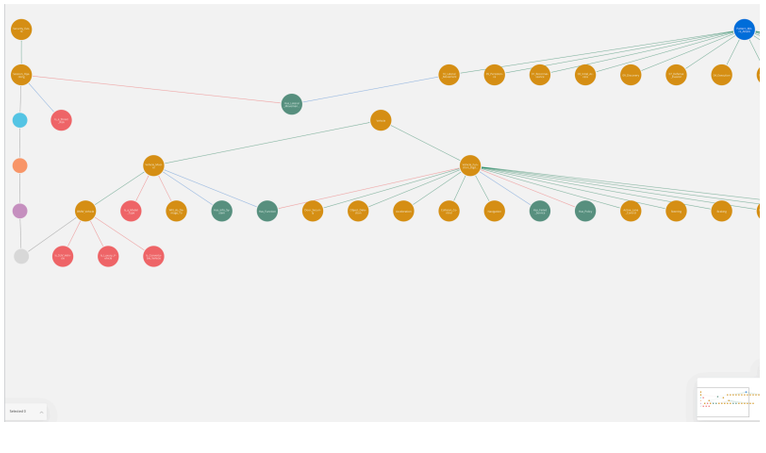
4) Integrate with MS Azure IOT Hub and MS Sentinel
Monitor physical device state, events and thresholds through the MS Sentinel security platform

[Steps 3 and 4 are not yet covered in this POC]

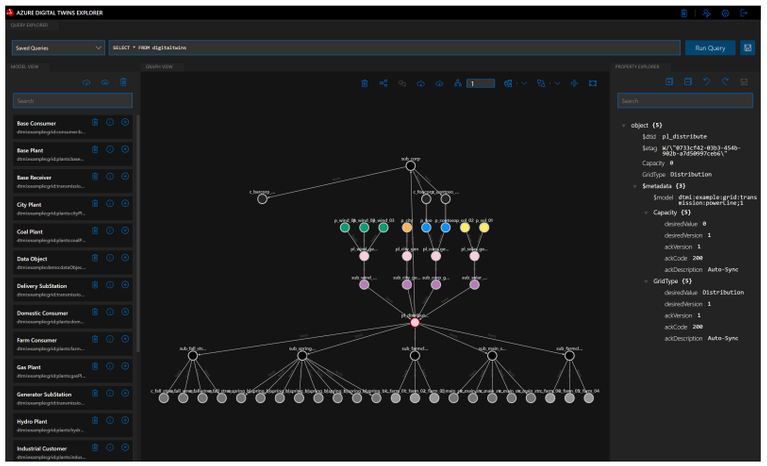
1



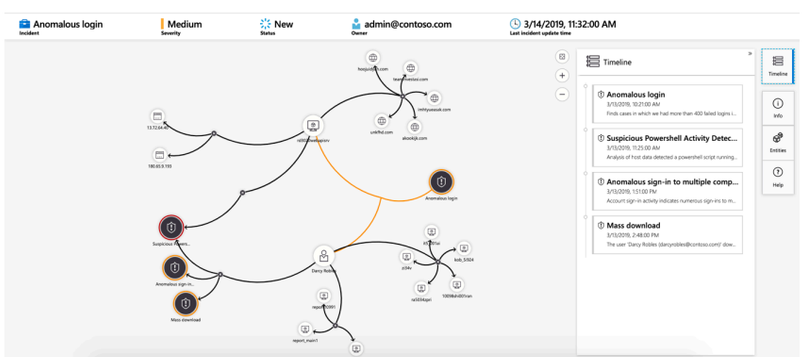
2



3



4



Smart City Cyber Security

For more information please visit

www.smartcitysecurity.net

https://smartcitysecurity.net/digital-twin-models/

https://smartcitysecurity.net/smart-city-cyber-security-framework/