Adaptive Defence of the Internet of Things (IoT) using the Belief-Desire-Intention (BDI) Model for Social Robots

Laura Rafferty
Faculty of Business and IT, Ontario Tech University, Canada
Laura.Rafferty@ontariotechu.ca

Áine MacDermott
School of Computer Science and Mathematics, Liverpool John Moores University, Liverpool, UK
a.m.macdermott@ljmu.ac.uk

Abstract

The Internet of Things (IoT) continues to introduce unique challenges and threats to cybersecurity. In parallel, adaptive and autonomous cyber defence has become an emerging research topic leveraging Artificial Intelligence (AI) for cybersecurity solutions that can learn to recognize, mitigate, and respond to cyber-attacks, evolving over time as the threat surface continues to increase in complexity. This paradigm presents an environment strongly conducive to agent-based systems, which offer a model for autonomous, cooperative, goal-oriented behaviours which can be applied to perform adaptive cyber defence activities. This paper presents a modular applied framework to leverage data models, domain knowledge, and multi-agent architecture to perform adaptive cyber defence capabilities through contextual policy generation and enforcement. The Belief-Desire-Intention (BDI) model is extended for behavioural modeling of agents to perform practical reasoning and deliberation of actions in pursuit of goals in an IoT environment with social robots.

Keywords: Cybersecurity, Internet of Things, Agent-Based Modelling, Adaptive Defence, Social Robots.

1. Introduction

Internet of Things (IoT) devices with increased connectivity and varying device capabilities are becoming more prominent in an array of industries, ranging from critical infrastructure, enterprises, automation, and healthcare. Within the consumer market, IoT devices within the household - 'smart home devices,' introduce a unique environment with sensitive personal data, availability requirements, limited expertise of users, and unique security threats. A fundamental problem with the interconnection of this 'Internet of Things' is that the IoT is creating a wider attack surface, with billions of new and emerging devices (MacDermott et al. 2018). Securing IoT devices becomes more complex due to the environmental complexity, the volume of data, expanding attack surface, and the plethora of avenues that can be exploited and targeted by malicious actors.

Artificial Intelligence (AI), machine learning, and automation are increasingly being adopted for cybersecurity applications as the intelligence and analytics gathered from environments can be leveraged and fed information from monitoring entities and cybersecurity frameworks. This increased integration of data for cybersecurity capabilities, also known as 'adaptive cyber defence', aims to create semi-autonomous cyber defences that can learn (based on observations in data and reference models) to recognize and respond to cyber-attacks, discover and mitigate weaknesses while evolving over time in response to changes in attacker behaviour, system health and readiness, and natural shifts in user behaviour (Marriott et al., 2021). Some limitations of current works within this domain include standalone solutions that are theoretical or do not provide interoperability. There is a strong need for a practical framework for the implementation of these capabilities, not only for security but also for regular IoT services (Coulter and Pan, 2018; Savaglio et al. 2021). Agent-based cybersecurity is an emerging topic in this interconnected paradigm, with agents being able to share information, make decisions based on observations, and have flexible hierarchies depending upon the needs of the infrastructure.

Our work aims to bridge the gap between theoretical multi-agent systems research and cybersecurity domain knowledge to provide a novel applied framework for adaptive cyber defence that can address a wide range of challenges and provide a foundation for significant future research in systems modelling for cybersecurity and other applications. We present a model smart home scenario containing several consumer IoT devices including, lighting, physical security, temperature control, and a social robot which participate in a network of Belief-Desire-Intention (BDI) software agents to perform security functions.

URI: https://hdl.handle.net/10125/106595
978-0-9981331-7-1

Within our framework, processes are modelled as multi-agent plans and tasks, where agents work together to achieve common goals to defend the network. We define a multi-agent adaptive cyber defence model within IoT smart home environments, using BDI agents to perform autonomous and adaptive goal-based reasoning for defence actions enabled by cybersecurity domain knowledge graphs.

The outline of this paper is as follows: In Section 2 we present background information and related works within the area. Section 3 outlines our Multi-agent System (MAS) architecture using BDI, and Section 4 details our Knowledge Graphs for BDI Agent Reasoning. In Section 5 we present a use-case scenario within an IoT smart home environment, and we conclude our findings in Section 6.

2. Background and related works

The rapid growth of the IoT has made a prominent impact on almost all areas of modern life, introducing a network of physical devices endowed with embedded sensors and networking capabilities to enable a vast array of pervasive services. The IoT is based on existing and evolving interoperable information and communication technologies. The development of IoT and smart home technologies are driven by advances in embedded and ubiquitous communication, virtualisation, and data analytics (Gafurov and Chung, 2019).

Smart home technologies are some of the most widely used and deployed applications for consumer IoT solutions, which provide digital services throughout the home through a range of networked devices (Hassija et al. 2019). Smart homes introduce an environment where IoT exists in the context of everyday objects in homes, such as doorbells, lighting, and fridges, and allow for greater automation and comfort of daily activities. These activities can be personalized, automated, and contextual services based on user input into the device controller or previously observed behaviour, such as playing the local weather report when the user turns off their Alexa alarm clock. Similarly, in assisted living or healthcare IoT scenarios (MacDermott, 2019), an alert can be sent to a healthcare provider if an individual is displaying abnormal behaviour symptomatic of a health issue.

Data available to smart home systems can be of volunteered, observed, or inferred types. Volunteered data is explicitly provided through the user in terms of profile preferences. Observed data is collected through sensors such as microphones or usage data. Finally, inferred data refers to information that has been correlated between volunteered and observed data, such as what time a user is likely to return home based on previous usage patterns. Users may be unaware of observed or inferred data collected and stored by the system, and this information can become very personal, such as behaviour and life patterns (Rafferty, 2022).

Due to the personal value of the data collected and retained by smart home systems, such data can be a target for attackers for a variety of reasons. As sensors are integrated into 'things' within the household, collected data can frequently be equated to physical observations, which can be further correlated with information collected from other sensors and sources. As IoT and smart homes are typically connected, other devices on the network, including smartphones and wearable devices, can interact with each other and share data. This makes it possible for further correlation across devices and for the data to be shared externally. Information collected can become increasingly intimate, such as health information, and can be correlated with data collected from other devices for further context extraction. Therefore, the privacy of all individuals within the home is at risk, including children who may be the primary users of some IoT technologies in the home, such as smart toys and social robots.

Agent-based Modelling (ABM) is a method of modelling systems composed of autonomous decision-making entities, known as agents, interacting with each other and their environment (Bonabeau, 2002). Agents execute actions based on a set of rules and often operate within an environment with other agents, known as Multi-Agent Systems (MAS). ABM has been used to simulate complex decentralized systems of autonomous agents to predict global system outcomes based on local interactions. Within this work, we apply agent-based modelling to IoT home scenarios. In ABM, the key attributes are the agent, environment, and relationships which can be defined as:

- **Agent:** An autonomous entity that makes decisions and actions based on a set of rules based on independent goals and perceptions.
- **Environment:** The physical or logical environment shared by all agents in a system, containing artifacts that can be perceived and impacted by agent actions.
- **Relationships:** The rules through which agents interact with each other, work together, or resolve conflicts.

The rules for how agents make individual decisions and interact with each other are formally defined in their model. The BDI model, originally developed by Bratman (1987), is used for behavioral modelling of agents to perform practical reasoning and the process of deciding what actions to perform to reach a goal. Agents receive sensory input through perceptions that influence their beliefs and implement their intended behaviors (intentions) to achieve desired states based on these beliefs, as illustrated in Figure 1.

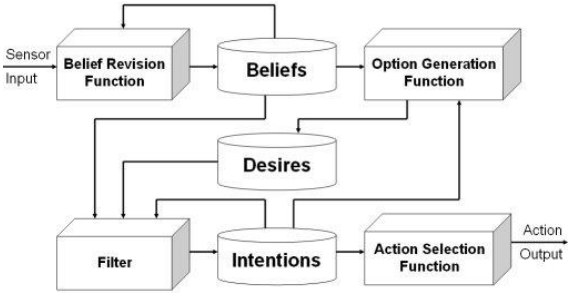

Figure 1. BDI components (Nunes et al., 2011)

The main components and functions of BDI can be modelled as (Simari and Parsons, 2011):

- **Percept** $P_n = \{p_0, p_1, \ldots, p_n\}$ represents a set of perceptions p taken as input by agents.
- **Beliefs** $B_n = \{b_0, b_1, \ldots, b_n\}$ represents a set of the information b maintained by the agent on its internal state and the environment states, updated according to each perception p.
- **Desires** $D_n = \{d_0, d_1, \ldots, d_n\}$ represents a set of the agent's goals to be achieved d, including properties and costs associated with each goal.
- **Intentions** $I_n = \{i_0, i_1, \ldots, i_n\}$ represents an action plan providing a set of states i the agent intends to bring about. The set of intentions must be consistent and not contain any conflicts.
- **Belief Revision Function** $BRF(p_n,B_n) \rightarrow B_m$ takes a perceptual input p_n and the agent's current belief set B_n, and determines a new set of beliefs B_m. Belief revision can include the following:
    - **Expansion**: a new sentence b_{n+1} is added to the belief set B.
    - **Revision**: a new sentence b_{n+1} that is inconsistent with a belief set B is added, but to maintain consistency with the resulting belief set, some old sentences are deleted.
- **Option Generation Function** $OPG(B_n,I_n) \rightarrow D_n$ determines the possible alternatives (desires) available to an agent-based on its current beliefs and intentions.
- **Filter Function** $FIL(B_n,D_n,I_n) \rightarrow I_n$ determines a consistent set of intentions based on the agent's current beliefs, desires and intentions.
- **Action Selection Function** $ACT(B_n,I_n,) \rightarrow \{a_0, a_1,\ldots,a_n\}$ implements means-ends reasoning to map the current set of beliefs *B* and intentions to a sequence of actions *a*.

A high-level process of the BDI model is conveyed in Figure 2. It is important to note that the BDI model does not account for dynamic plan generation and instead depends on a predefined plan database. While this approach is commonly static and has limitations to scalability and adaptation to evolving collections of knowledge, we expand on this approach by introducing novel integration with knowledge graphs – explained in Section 4.

```
B := B0;
    I := I0;
    while true do
            get next percept p;
            B := BRF(B,p);
            D := OPG(B,I);
            I := FIL(B,D,I);
            N := PLN(B,I);
            execute( )
    end while
```

Figure 2. BDI process description

Agent-based approaches have also been applied to smart home environments. Kravari and Bassiliades (2019) introduce shared aims of characteristics between IoT, multi-agent systems, and microservice architecture due to their distributed, autonomous, collaborative, and goal-oriented nature. The authors apply this approach through a novel reputation-oriented trust model to support the challenge of intelligence and trustworthiness of IoT. Further, Rafalimanana et al. (2020) adopt a collaborative agent-based approach to create a link between AI and services in IoT. The authors pair BDI-agents with Representational State Transfer (REST) service technologies to exploit the agent capabilities as a service. Hilal and Basir (2014) propose an agent-based sensor management architecture for pervasive surveillance to support the coordination of sensor nodes and maintain situational awareness of the environment. The approach combines the advantages of holonic, federated, and market-based coordination architectures and models each node as an intelligent sensor using BDI.

In addition, recent efforts of the North Atlantic Treaty Organization (NATO) to develop an Autonomous Cyber-Defence Agent (AICA) reference architecture (Théron et al., 2014) have been a significant development in the literature toward an autonomous agent system for cyber defence. While not specific to IoT or BDI agents, the focus of NATO's work is to enable future defence actions on largely autonomous military assets where human intervention may not be possible. The architecture provides capabilities for autonomous planning and execution of multi-step activities, adversarial reasoning in response to intelligence, and the ability to remain undetected. While the existing works demonstrate promising directions in applications of agent-based approaches to cybersecurity, they have focused on specific security capabilities rather than on the problem from a holistic point of view.

3. Multi-agent system (MAS) architecture using BDI

This section presents our proposed MAS architecture, which integrates autonomous defence capabilities into adaptive intelligent software agents situated to respond to the evolving cybersecurity threat landscape. Processes are modelled as multi-agent plans and tasks, where agents work together through a control and coordination hierarchy to achieve common goals to defend the network according to security requirements. Agents use the BDI model to perform multi-agent goal-based deliberative reasoning for defence actions which are informed by domain knowledge graphs. The MAS architecture is composed of 3 main components: '*Security Services*,' '*Coordination*,' and '*Mission Deployment'* into the IoT environment. The high-level system architecture is shown in Figure 3, illustrating how the core model interacts with the IoT environment. Within each of these layers, agents perform operations, communicate with each other, and make use of the available resources throughout the system.

In the '*Security Services'* layer high-level security decisions are made by control agents, based on system observations, security requirements, and domain knowledge graphs. This data is used to generate defence policies to be actioned by agents throughout the network. The results of these policies are utilized by the '*Coordination Layer*,' where coordination agents take the defence policy as input, identifying security goals to be achieved and then mapping them to actions with corresponding functions for prioritization. These goals and actions are planned and prioritized through workflow planning and coordination of available resources to generate subsequent missions, which are monitored by mission control – *'Mission Deployment'*. Each mission consists of an action set, goal(s), prospective utility, and a set of agents with predefined beliefs, desires, intentions, and roles. Agents deployed through missions interact with each other in an agent collaboration environment as well as directly in the IoT environment to perform actions to achieve their mission objectives. An agent can interact with different components within the IoT environment, including devices, applications, and cloud services, either through API or directly hosted within the resource.

3.1. Agent hierarchy

A hierarchical agent structure is used for organization-level insights and emergent behavior in agents across the network to achieve holistic security goals. In this section, we continue to expand on our model for the generation of agents to perform the actions selected by the controller. Section 3.2 describes the behaviours of the MAS controller (Level 0) as a strategic defensive BDI agent which generates defence policies (intentions) based on beliefs and desires for an ideal security state. As shown in Figure 4, this results in a hierarchical BDI agent structure.

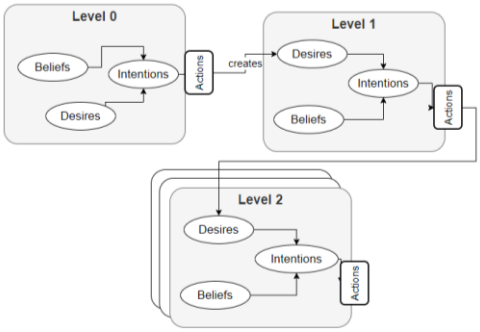

Figure 4. BDI hierarchy

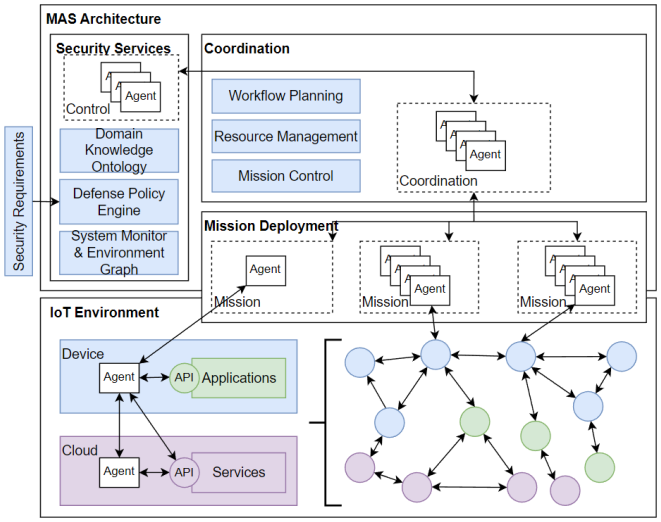

Figure 3. High-level MAS architecture overview

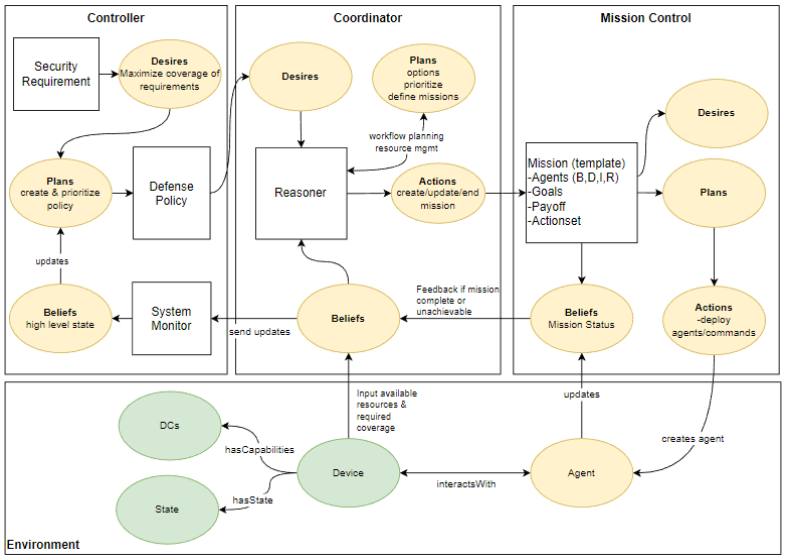

Figure 5. Multi-agent BDI inheritance of policy desires

The coordinator generates a high-level intention which is the template for the creation of sub-agents (Level 1+) with corresponding desires. Coordination mechanisms allow coordination between the controllers and coordinators to allocate resources according to the requirements. Taking as input the profile of the defenders θ, attack types a, and resources available k, a coordination mechanism function $\pi : (\theta, k, a) \rightarrow (x, t)$ is generated, which outputs a strategy x for the target t.

While all decisions and sensory aspects of the system are performed by BDI agents of different functions, Figure 5 shows how initial security requirements are inherited as desires by downstream agents with the ability to perform required actions accordingly, where the security requirements are first received by the controller to create the defence policy according to the knowledge of the environment from the system monitor. The defence policy translates into agent desires, in which the coordinator performs mission generation through planned workflows and resource management. Missions are created to assign associated desires and functions to capable agents deployed within the environment to achieve the overall desires.

3.2. Requirements definition

Modeling the environment in a way that can be understood and reasoned by the agents is critical to situational awareness and understanding of environment states for the agents to act upon. In a common format, security requirements can be defined by industry standards, vendor policies, and user preferences. These requirements are combined with observations from the system monitor and domain knowledge graph to generate context-aware policies with respect to availability, coverage, and exposures. Together these form the defence policy used by the MAS control agent(s) to generate security goals and corresponding plans. If any changes are made to the requirements, the defence policy will be updated as necessary.

Table 1 shows the elements of the baseline security requirement definition format. The security requirements can be in any common format, such as JavaScript Object Notation (JSON) or eXtensible Markup Language (XML) - JSON was selected for demonstrative purposes. Figure 6 provides an example of the Vulnerability Management requirement, which is applied globally to all devices on the network.

Table 1. Baseline security requirements format

Field	Data
ID	int
Name	string
Requirement	string
Priority	[1-5]
Associated platforms	Global |Group | Device Profile | Device
Desired state	(“Subject”, “State”)
Associated Security Properties	Confidentiality | Integrity | Availability | Non-Repudiation | Authenticity | All
Associated Device Capabilities	DC […]

```
"ID": 004,
"Name": "Vulnerability Management",
"Requirement": "Identify and eliminate known vulnerabilities"
"Priority": 1,
"Associated Platforms": "Global",
"Desired State": "Device Version = Up to Date",
"Associated Security Properties": "All",
"Associated Device Capabilities": ["DC5.1", "DC5.2", "DC5.3"]
```

Figure 6. Security requirement example - vulnerability management

The requirement includes the associated device capabilities which can achieve it, as well as the desired state of "*Device Version is Up to Date*," which will define the target state for the agents' desires. Our knowledge graphs provide a data model for security requirements to be interpreted by a policy engine according to the context of the environment and inferences to cybersecurity domain knowledge. The policy engine generates the policies by validating the security requirements provided as input. Through the hierarchical agent model, multiple layers of policy types can be maintained.

3.3. Coverage monitor and exposures

The Coverage Monitor tracks security controls in place in relation to security requirements and compensating controls for exposures and attacks. This is leveraged by the controller to track overall coverage to inform the policy prioritization. While some devices may be required to comply with a certain security requirement without having corresponding device capabilities to achieve it, compensating controls will need to be put in place. Once the controller has situational awareness of the limitations and capabilities of the network, appropriate missions can be deployed to provide coverage.

The Exposure Monitor tracks known vulnerabilities and configuration risks within the network, as well as an understanding of the risk associated with the exposure. The domain knowledge ontology is used for enriching exposure data based on Common Vulnerability Scoring System (CVSS) (NIST, 2023). While exposures are known vulnerabilities that have not been exploited, if an attack is detected targeted in the exposure, it would be listed in the attack monitor. Each exposure should be prioritized by the controller for coverage according to the level of risk. The exposures monitor contains the following fields:

- **Type:** the type of exposure as Common Vulnerabilities and Exposures (CVE) or risk.
- **CVE ID:** the CVE associated with the exposure.
- **Risk Score:** as listed in CVE.
- **Impact:** the impact of the exposure as listed in CVE (confidentiality, integrity, or availability).
- **Exploitability:** as listed in CVE (privileges required, attack vector, user interaction, scope).
- **Associated Assets:** the assets affected by the exposure.
- **Status:** tracking the status to indicate whether the exposure is active or remediated.

3.4. Alerts

The Alerts Monitor tracks alerts indicating suspicious or malicious behavior on the network to be investigated and/or remediated. Alerts are created within the monitoring controls and analytics, and contain the unique ID for the alert, impacted assets, level of risk determined by the domain knowledge graph, and MITRE ATT&CK data associated with the alert (MITRE, 2023). Other associated information includes the data source telemetry that the alert originated from, the type of platform associated with the alert, and data model references (object, actions, and fields associated with the cyber analytics data model).

4. Knowledge graphs for BDI agent reasoning

Our graph architecture integrates three separate layers for context into the network environment, cybersecurity domain knowledge and BDI agent knowledge, as shown in Figure 7. The '*Environment'* layer is used to model the devices and entities within the network to provide ongoing context and state awareness. The '*BDI Agent*' layer is used to model agent planning, actions and workflows based on knowledge of device capabilities, security requirements and context from the other two layers. The '*Cyber Domain Knowledge*' layer integrates industry frameworks into a common model for identifying vulnerabilities and exposures, inferring security risks, attack detections and analysis, and informing applicable defence techniques based on policies and environmental awareness. Each layer is highly related to informing agent decisions based on knowledge of the environment states and capabilities.

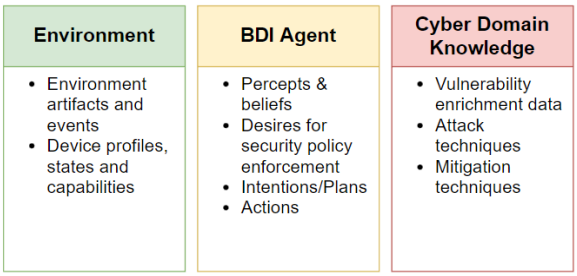

Figure 7. High-level graph layer interactions

Cybersecurity domain knowledge is used to support actions based on risk profiles of the known environment and security requirements. Figure 8 presents an overall view of the relationships between each graph component of the environment, agent, and domain knowledge.

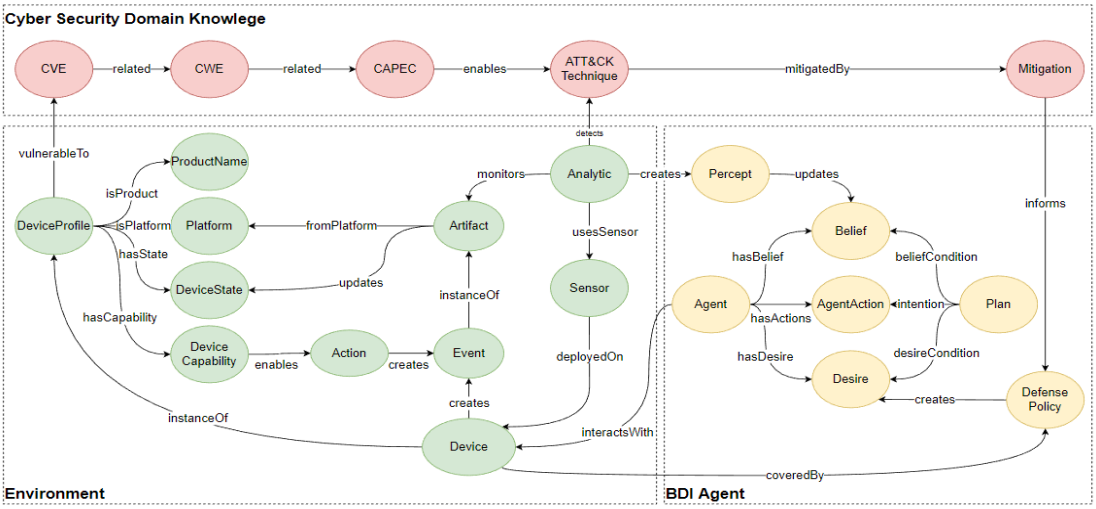

Figure 8. Knowledge graph meta model

Maintaining ongoing knowledge of the environment is a critical function for rational agents to interact and receive timely feedback on their environment states. The environment graph is the basis of the agents' situational awareness and is further augmented by domain knowledge enrichment data to infer the security implications.

In our scenario, BDI modelling can be used as a rational model to add proactive behaviours. Using a graph model, key components of the environment can be defined, categorized, labelled, and related using a data model that allows for interconnectivity with domain knowledge graphs and agent planning. The environment graph has been designed with the following key requirements and integrations in mind:

1. Model environment for agents to interpret devices, attributes, states, capabilities, and possible actions.
2. Attributes to be mapped to security domain knowledge for understanding vulnerabilities, risk analysis, relation to security requirements/policies.
3. Flexible reference data profiles and maintenance.

4.1. Cypher queries for agent functions

Modeling available actions based on device capabilities and belief states allows an agent to query the graph for a plan of action to achieve a path to a target's desired state. BDI agents can leverage the graph to support their belief revision, plan selection, and action selection functions using Neo4j (2022) cypher queries.

Table 2. BDI graph node descriptions

Node Label	Properties	Relationships
Percept	value	Belief
Belief	value	Percept, AgentAction
Desire	value	State/Belief, Plan
AgentAction	value	DeviceCapability, Belief

Belief revision is triggered by an agent receiving a new percept. The agent graph supports the agent's belief revision function by providing relationships between types of percepts and the beliefs to be inferred. When a new desire is obtained, such as through a new security policy or mission, the agent must retrieve an appropriate plan from the graph to pursue the desired state.

The process of plan selection (along with the applicable cypher query), to return a plan in the form of belief-action pairs is as follows:

1. Find a state/belief that is the objective of the active desire
2. Find *a* actions that the target state is "achieved by" (i.e., download update, install update)
3. Find *b* beliefs that precede these as "next action" (i.e., update available, update downloaded)
4. Build belief/action pairs based on *b* and *a*

```
MATCH (:Desire {value: "$desire"}) -
[:OBJECTIVE]-> (targetbelief)
MATCH (targetbelief) -[:ACHIEVEDBY]->
(selectedActions)
MATCH (preBeliefs) -[:NEXTACTION]->
(selectedActions)
RETURN targetbelief, selectedActions,
preBeliefs
```

Based on a plan to achieve a particular desired state, an agent must select an appropriate next action. The below query is used to return the next action based on the agent's current beliefs:

```
MATCH (:Desire {value:"$desire"}) -
[:OBJECTIVE]-> (targetbelief)
MATCH (targetbelief) -[:ACHIEVEDBY]->
(selectedAction)
MATCH (preBeliefs:Belief{value:"$belief"})-
[:NEXTACTION]-> (selectedAction)
RETURN selectedAction
```

5. Smart home scenario implementation

Next, we present our MAS architecture for adaptive cyber defence in a smart home network. Our implementation architecture consists of three components: Colored Petri Nets (CPN), knowledge graph database, and the simulation engine. The implementation simulates a fictional smart home environment, as shown in Figure 9 as an illustrative example of a realistic use case.

The scenario illustrates a single-bedroom apartment that contains a variety of IoT devices for physical security, assisted living – a social robot, temperature control, lighting, entertainment, personal devices, and network devices. The devices have been added to the Neo4j environment graph and basic functions created within the simulation engine. Device capabilities for each device according to NIST have been generated (NIST, 2023). We simulated our environment and agent instances in CPN Tools to visualize the states and transitions as messages are sent through the network. For validation of basic agent BDI reasoning and communication, our implementation used external integrations with Neo4j and Simulation engine as the basis for larger scale intelligence and knowledge reasoning. While the CPN walks through all the logical steps of the environment and agent algorithm and visualizes the data at each step within the corresponding place and transition, the actual intelligence and processing take place outside of the net through the functions called to the simulation engine integration to execute the corresponding functions and execute cypher queries to the Neo4j graphs accordingly. We defined a common message format for simple simulated network communications within the system. A colorset of "MSG" has been defined consisting of the following fields: msg = { src="", dst="", kind="", data=[] }, where **Src:** the source of the message, **dst:** the destination of the message, **Kind:** identifies the type of message to handle data fields, **Data:** a string list consisting of the actual data of the message within the appropriate fields.

The "kinds" of messages described in Table 3 have been defined to pass messages across the network and for internal messages between the agent and host devices. Figure 10 implements the BDI components for the given scenario, generating Belief Revisions, Perceptions, and Reasoning based on the values shared among monitoring entities. Each device in the scenario has modelled device capabilities and security requirements which are reflected in the knowledge graph, while each agent has a role it plays to achieve these requirements within the ecosystem. For each observation a decision is made using the Reasoner engine and an action passed to the Agent.

Table 3. Message "kind" definitions

Kind	Data Format	Usage
notify	[("value","UpdateNotification"), ("ver", "$ver")]	For update notifications
request	Same as action	Request msg to another device
response	[("Function name","id"),("status", "Success/Failed"),(result/error))]	Response msg to another device
action	(Function name, ID),(Function parameters) [("update", "id"), ("ver", "$ver")]	Agent action on host device
Action Response	[("Function name","id"),("status", "Success/Failed"),(result/error))]	Host device response to action successful/failed

Device Listing	
Type	**Product Name**
Physical Security	-Security System -Smart Lock
Social Robot	-Robot1
Lighting	-Light Hub -Light[1-4]
Temperature Control	-Thermostat -Temp Sensor[1-2]
Audio/Video	-Smart TV -Smart Speaker
Personal Devices	-Laptop -Smart Phone
Network & Control	-Router -IoT Hub

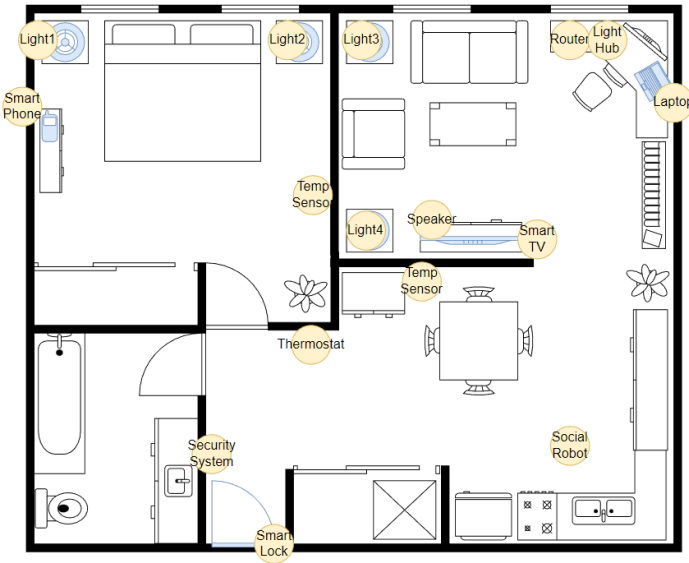

Figure 9. Model smart home environment and device listing

Our knowledge graphs have been created to represent BDI relationships for each of the security requirements and corresponding device capabilities related to each device as applicable within the environment graph. Agents can leverage the graph for belief revision and planning to select appropriate actions based on their beliefs and desires. Based on the security requirements, a defence policy is created by the controller according to the knowledge of the environment, which defines the high-level agent's desires for the system. The elements of the defence policy are input into the knowledge graph, where the goal of each policy is identified as an agent "desire" associated with the desired system state. Each desire is related to each device or group of devices to which the policy is to be applied. After the defence policy is defined and passed on to the coordinator, it must take the appropriate actions to ensure the policy is enforced. The hierarchical agent model has been designed to limit unnecessary communications between agents while allowing autonomous agent behavior with distributed control and knowledge. Leveraging reusable agent templates and control hierarchies, we can limit unnecessary agent calls to the Neo4j graph by retaining appropriate plans within agent memory after they are deployed. Through this design, the control and coordination agents will perform most of the Neo4j requests to the environment and BDI graphs. In contrast, most of the decisions and queries would take place during the initialization of the system when the defence policy is being created and missions are coordinated.

The simulation engine was developed for the purpose of simulating the device and agent instances for demonstrating our model, where the functions to receive and process messages to perform basic actions are defined in the graph. While the actions and functions are not performed within CPN tools directly, the simulation engine parses the message format to perform the appropriate functions and update states accordingly.

While the cybersecurity capabilities in this scenario are largely isolated from operational functions of the IoT devices, our model can be integrated further. For example, social robots are designed to interact and communicate with humans and other robots (Thalmann, 2022), following social behaviours and rules attached to their role for applications such as companionship, customer service, tutoring. Social robots can interact with humans or other autonomous agents, devices, or smart 'things' by following defined social behaviours and rules. BDI for social robots can be used as a rational model to add proactive behaviours and can measure how they behave in certain scenarios (K.C, 2019). For social robots, the sensor outputs can build the belief sets to signify the environment around the robot based on the defined scenario and context. A particular belief set, as explained above, describes a specific situation in which the robot is located in an instance of time. Based on environmental factors such as date/time/location, goals can be defined for the robot, known as a desire. This data can be used to map and measure the behaviours in a home environment and be used as an inference model on mannerisms and actions of all devices.

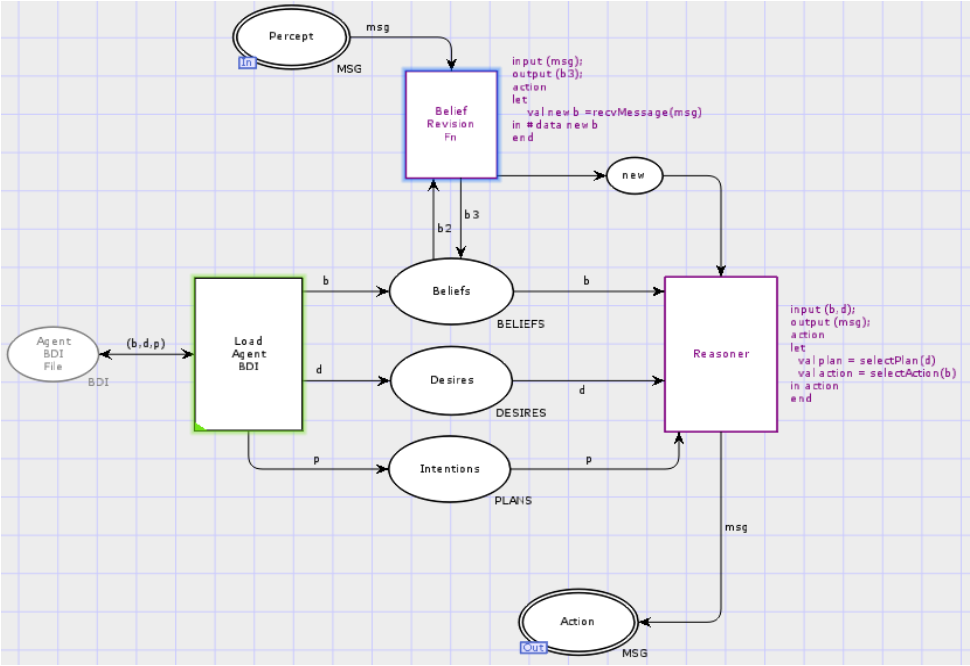

Figure 10. Agent CPN

6. Conclusion

This paper presents a multi-agent architecture for adaptive cyber defence with an agent reasoning model, control and coordination hierarchy. We created a novel extension of the BDI model enabled by knowledge graphs for cyber modelling based on industry knowledge bases which can be leveraged for policy-based, adaptive agent reasoning. IoT devices are increasingly being utilized within workplaces and home environments, introducing novel security concerns and technical deployments which can benefit from adaptive cyber defence. While our MAS architecture and BDI characteristics are unique, this paper demonstrates only a subset of the features of the approach. More complex use cases will build upon the findings within and replicate real-world scenarios.

This provides the foundation for future works on agent-based solutions for continued development, application, and optimization to support the advancement of autonomous, adaptive cyber defence. Further, the model can also be extended to additional use cases applicable to social robots, smart home, and other scenarios that can make use of policy-based, adaptive agent reasoning.

7. References

Bonabeau, E., 2002. Agent-based modeling: Methods and techniques for simulating human systems. Proceedings of the national academy of sciences, 99(suppl_3), pp.7280-7287.

Coulter, R. and Pan, L., 2018. Intelligent agents defending for an IoT world: A review. Computers & Security, 73, pp.439-458.

Bratman, M.E., 1987. Intention, Plans, and Practical Reason. Cambridge. Harvard University Press.

Gafurov, K. and Chung, T.M., 2019. Comprehensive survey on internet of things, architecture, security aspects, applications, related technologies, economic perspective, and future directions. Journal of Information Processing Systems, 15(4), pp.797-819.

Hilal, A.R. and Basir, O.A., 2014. A scalable sensor management architecture using BDI model for pervasive surveillance. IEEE Systems Journal, 9(2), pp.529-541.

Hassija, V., Chamola, V., Saxena, V., Jain, D., Goyal, P. and Sikdar, B., 2019. A survey on IoT security: application areas, security threats, and solution architectures. IEEE Access, 7, pp.82721-82743.

KC, U. and Chodorowski, J., 2019. A case study of adding proactivity in indoor social robots using belief–desire–intention (BDI) model. Biomimetics, 4(4), p.74.

Kravari, K. and Bassiliades, N., 2019. StoRM: A social agent-based trust model for the internet of things adopting microservice architecture. Simulation Modelling Practice and Theory, 94, pp.286-302.

MacDermott, A., Baker, T. and Shi, Q., 2018, February. IoT forensics: Challenges for the IoA era. In 2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS) (pp. 1-5). IEEE.

MacDermott, Á., Kendrick, P., Idowu, I., Ashall, M. and Shi, Q., 2019, June. Securing things in the healthcare internet of things. In 2019 Global IoT Summit (GIoTS) (pp. 1-6). IEEE.

Marriott, D., Ferguson-Walter, K., Fugate, S. and Carvalho, M., 2021. Proceedings of the 1st International Workshop on Adaptive Cyber Defense. arXiv preprint arXiv:2108.08476.

MITRE, 2023. MITRE Corporation, "CVE Program Mission," CVE, 2022. Available at: https://cve.mitre.org/

Neo4j Inc., 2022. "Neo4j Graph Database," Neo4j Graph Database. Available at: https://neo4j.com/product/neo4j-graph-database

NIST, 2023. Vulnerability Metrics, National Vulnerability Database (NVD). Available at: https://nvd.nist.gov/vuln-metrics/cvss#:~:text=The%20Common%20Vulnerability%20Scoring%20System,Base%2C%20Temporal%2C%20and%20Environmental.

Nunes, I., De Lucena, C.J. and Luck, M., 2011. BDI4JADE: a BDI layer on top of JADE. In Ninth International Workshop on Programming Multi-Agent Systems (ProMAS 2011), Taipei, Taiwan (pp. 88-103).

Patel, K.K., Patel, S.M. and Scholar, P., 2016. Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. International journal of engineering science and computing, 6(5).

Rafalimanana, H.F., Razafindramintsa, J.L., Cherrier, S., Mahatody, T., George, L. and Manantsoa, V., 2020. Jason-rs, A collaboration between agents and an IoT platform. In Machine Learning for Networking: Second IFIP TC 6 International Conference, MLN 2019, Paris, France, December 3–5, 2019, 2 (pp. 403-413). Springer.

Rafferty, L., 2022. Agent-based modeling framework for adaptive cyber defence of the Internet of Things (Doctoral dissertation).

Roman, R., Zhou, J. and Lopez, J., 2013. On the features and challenges of security and privacy in distributed internet of things. Computer networks, 57(10), pp.2266-2279.

Savaglio, C., Ganzha, M., Paprzycki, M., Bădică, C., Ivanović, M. and Fortino, G., 2020. Agent-based Internet of Things: State-of-the-art and research challenges. Future Generation Computer Systems, 102, pp.1038-1053.

Simari, G.I. and Parsons, S.D., 2011. Markov Decision Processes and the Belief-Desire-Intention Model: Bridging the Gap for Autonomous Agents. Springer Science & Business Media.

Thalmann, N.M. (2022). Social Robots: Their History and What They Can Do for Us. In: Werthner, H., Prem, E., Lee, E.A., Ghezzi, C. (eds) Perspectives on Digital Humanism. Springer, Cham. https://doi.org/10.1007/978-3-030-86144-5_2

Théron, A., Drašar, P., Dushku, M., LeBlanc, et al. 2018. Autonomous Intelligent Cyber-defense Agent (AICA) reference architecture. release 2.0. arXiv preprint arXiv:1803.1