mathematics

MDPI

Article

Formal Model of IDS Based on BDI Logic

Ján Perháč *,†, Valerie Novitzká †, William Steingartner † and Zuzana Bilanová †

Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovakia; Valerie.Novitzka@tuke.sk (V.N.); William.Steingartner@tuke.sk (W.S.); Zuzana.Bilanova@tuke.sk (Z.B.)

* Correspondence: Jan.Perhac@tuke.sk

† These authors contributed equally to this work.

Abstract: Computer network security is an important aspect of computer science. Many researchers are trying to increase security using different methods, technologies, or tools. One of the most common practices is the deployment of an Intrusion Detection System (IDS). The current state of IDS brings only passive protection from network intrusions, i.e., IDS can only detect possible intrusions. Due to that, the manual intervention of an administrator is needed. In our paper, we present a logical model of an active IDS based on category theory, coalgebras, linear logic, and Belief–Desire–Intention (BDI) logic. Such an IDS can not only detect intrusions but also autonomously react to them according to a defined security policy. We demonstrate our approach on a motivating example with real network intrusions.

Keywords: BDI logic; linear logic; IDS; category theory; coalgebra

Citation: Perháč, J.; Novitzká, V.; Steingartner, W.; Bilanová, Z. Formal Model of IDS Based on BDI Logic. *Mathematics* **2021**, *9*, 2290. https://doi.org/10.3390/math9182290

Academic Editor: Radi Romansky

Received: 31 July 2021
Accepted: 11 September 2021
Published: 17 September 2021

1. Introduction

In today's information society, computer security is undoubtedly a very important area of research. The rapid technological development has brought the advent of personal computers, laptops, smart devices, the Internet of Things (IoT), etc., which means computers are already involved in every aspect of human life. Therefore, various sensitive and confidential information flows evermore through the network. This brings the necessity to secure that information from falling into malicious hands.

1.1. Motivation

One of the most common practices to increase the security of computer networks is the use of Intrusion Detection Systems (IDSs). Generally, an IDS is a device or software application that monitors a computer system or a network [1] for malicious activities. Our work is dedicated to network-based intrusion detection systems where the detection of intrusions is based on the patterns of known intrusions. The advantages of such an IDS are that, after the implementation, the system is immediately ready to detect intrusions, and it is reliable in the detection of known intrusions. On the other hand, a disadvantage is the inability to detect new intrusions [2]. IDSs are based on observing the usage of a computer network and the statistics learned so that intrusions can be detected. Their advantage is the ability to detect new intrusions, but the disadvantages are, e.g., the need for long-term observation of the computer network before a representative sample of statistics can be created. It can also miss old but well-known simple intrusions because these do not greatly disturb the traffic. However, these intrusion detection systems provide only passive protection from attackers. When an intrusion is detected, only the respective logging of the intrusion is performed, without any reaction to it. Therefore, after intrusion detection, a computer network requires the intervention of an administrator.

Another disadvantage of the current state of intrusion detection is that there is no exact formal description or verifiable model of IDSs [3]. Compared with other (more common)

approaches of software engineering, which require testing a system after any change in its model, an exact mathematical model can be useful to prove the correctness and to minimize the undesirable behavior of a system in the design phase, before its physical implementation.

We published a first attempt in this field in [4], but the proposed logical model dealt only with passive IDSs; therefore, it was only able to gain objective knowledge from a rational belief about an intrusion. An objective of this paper is to extend our work by designing a formal verifiable logical model for an IDS, which allows us to obtain not only a belief about an intrusion but also an active reaction to it. Such an IDS would not only autonomously react to a detected intrusion but also create countermeasures against it, and it would prevent future occurrences of similar types of intrusions.

1.2. Work Plan

For the passive part of our model [4], we used a coalgebra for a polynomial endofunctor (formally defined in Section 2.2.2) as an appropriate method to model a state-based system and to observe its internal state, and we used the expressive power and dynamics of linear logic (formally defined in Section 2.3.1). In this paper, we extend our logic to the linear BDI logic, mainly with the resource-oriented casual nature of linear connectives. This brings a new and stronger expressive power, allowing for the description of real-world properties.

BDI logic (formally defined in Section 2.3.2) is a logic of three modalities, *belief, desire, and intention*, about certain properties. This allows us to formulate the desired state of an environment, which has to be in symbiosis with its beliefs and can be altered by intentions, i.e., plans. BDI logic was originally developed for reasoning about BDI agents. There is no exact definition of a BDI agent. A common informal definition is that a BDI agent is some abstract autonomous unit with the following characteristic abilities:

- Able to perceive an environment in some way (through some perception devices, sensors, actuators, etc.);
- Able to process the received information and to make decisions based on it;
- Able to influence an environment.

The active part of our model consists of three units, which represent the mental attitude of a BDI agent as follows:

- *Belief*—represents obtaining a *belief* about an intrusion;
- *Desire*—represents the logically formulated security policy of a network;
- *Intention*—represents a database of plans for reactions to a breach in security policy in gaining a belief about an intrusion.

We demonstrate this model on the specific examples of three intrusions. This is described in Section 4.1.

1.3. Structure of the Paper

In Section 2, we present the basic notions used in the formal methods and our notation. In Section 3, we present our first contribution: the introduction of a linear belief–desire–intention logical system (linear BDI), its syntax, and its semantics. Then, we continue with our main result (Section 4), which is the design of linear BDI model for an active IDS. First, we convert the real IDS's network intrusion detection signatures to many-type signatures. We construct a network stream of packets as the state-based category of packets, and we model its behavior as a coalgebra for a polynomial endofunctor (a passive IDS). As the last step, we present the proposed model of the active IDS based on our defined linear BDI logic.

1.4. Related Works and Our Proposition

The logical model for an active IDS presented in this paper is based on linear logic and a BDI model. There are several works and ideas on how to use a BDI model in computer science.

Mazal et al. presented the formalism of Object-Oriented Petri Nets in their publication [5] by modeling BDI agents using their PNagent framework. They showed that Object-Oriented Petri Nets provide a user-friendly and intuitive graphical method for modeling beliefs, desires, and intentions using the agent interpreter.

The other possible usages of a BDI model in software engineering are numerous, e.g., in Nunes's publication [6], he proposed a model-driven approach to develop BDI agents able to select plans based on soft goals and preferences. Braubach et al. presented the JADEX system [7], which combines the advantages of an agent middleware with a reasoning engine. The modular approach was introduced by Dastani and Steunebrink in [8], where they presented their ideas to design and integrate modules in BDI-based agent programming languages.

Interesting ideas for the use of a BDI model in IoT, specifically in the area of smart houses, were presented by Qingquan Sun et al. in their publication [9]. They presented a multi-agent design framework for controlling smart house features and the automation of home applications. They proposed various techniques on how to develop distributed multi-agent sensor/actuator networks. They designed a BDI model for the individual behavior of an agent and a method for regulation policy. For a system evaluation, they also used a Petri Net-based method. Sangulagi et al. [10] presented an application of a BDI model to solve a problem with an information fusion in a wireless sensor network.

The application of a BDI model in computer security or networks is not widely proposed. A few approaches have been published, e.g., Boudaoud et al. presented a multi-agent system-based model for network security management in their paper [11]. They described a model of security policy management. Their approach is dedicated to the practical implementation of certain schemas of actions within select security policies. As an interesting idea, Lin et al. used a BDI model approach to eliminate human errors within a computer security policy in their publication [12].

Our approach presented in this paper introduces a new usage of a BDI model with a single agent system combined with a new logical system for increasing computer network security using the automated reaction of IDSs to malicious activities.

1.5. Contributions

A logical model presented in this paper is raised from our long-term effort to design an exact, formal verifiable model for an active intrusion detection system [3]. Our first contribution is the formulation of a linear BDI logical system. We define its syntax and express its semantics using Kripke's possible worlds method [13]. For that, we use a linear logic [14], a coalgebraic logic [15], and a BDI logic [16]. Our first works were in regard to an active IDS, published in our papers [17,18], where we defined a fragment of our logic, modified well-known IRMA architecture [19], and logically modeled the behavior of the proposed architecture on network intrusions.

Furthermore, in this paper, we turn to a formal specification of a passive IDS. Using a category theory approach, we modeled its behavior as a state-based transition system using a coalgebra for a polynomial endofunctor. Our main result is the design of an active IDS logical model based on the Belief–Desire–Intention agent philosophy. Based on that, we proposed a formal linear BDI logical model for an active IDS.

In terms of IDSs, it is possible to use BDI logic with the following cycle of the agent:

1. An IDS (a coalgebra-formalized IDS) observes a packet stream;
2. Knowledge about a possible intrusion could be obtained (if not, a coalgebra continues to check the next packet);
3. Based on that, an agent can gain a belief about it;
4. The agent confronts a belief based on their desires:

- If a gained believe is not against the agent's desires, the coalgebra continues to check the next packet, and
- If it is against their desires, an agent realizes actions through intentions (plans) to restore their desired state.

The principle of our goal is depicted in the Figure 1.

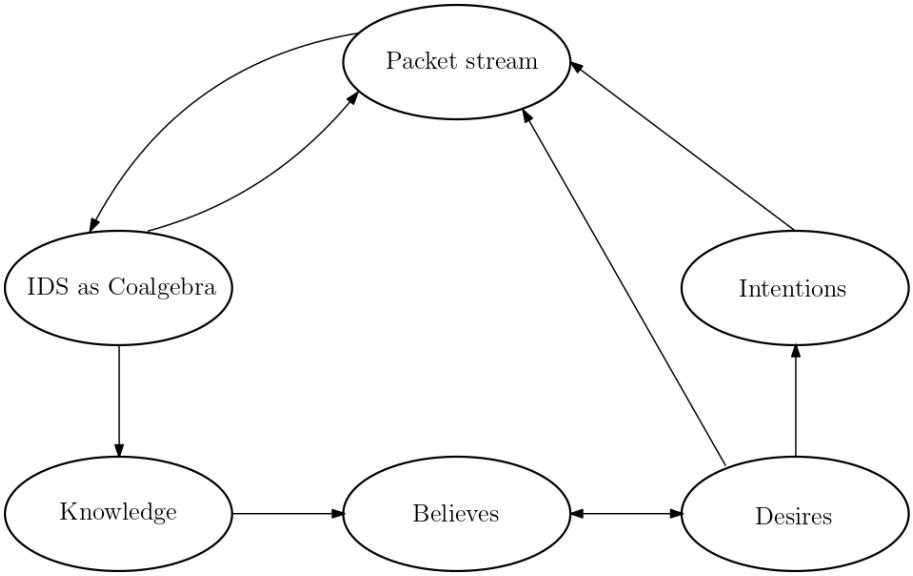


Figure 1. An agent's cycle.

2. Overview of the Methods Used

The methods used. such as category theory, linear logic, and BDI logic, are strict, exact, and mathematically proven. Linear logic and the proposed extensions presuppose the design of an appropriate deduction system that allows for logical reasoning and proves all of the designed properties. This guarantees the correctness and reliability of our IDS logical model before its implementation. In this section, we briefly present the necessary basic notions from methods used in our work from category theory and non-classical logical systems such as linear logic and BDI logic.

2.1. Many-Typed Signature

Many-typed signature is a well-known and important notion in the theory of algebraic specifications [20]. It is meant to declare the structure of a data type. To formally specify the structure of a packet, we formulate a signature $\Sigma = (T, F)$ for it. The class T contains the names of types in the algebraic specification, and class F encloses the function symbols (or operation specifications) over the types in T. Those operations can be constructors, deconstructors, and selectors. For our approach based on constructing the coalgebras, most of the selectors are important, as we use them to observe internal states of a state-based system in detail.

2.2. Category Theory

We focus on constructing the semantic model of IDS behavior as a category. This is because categories are mathematical structures that allow for expressing an environment of states. We consider particular states as category objects, and the dynamic of internal processes of IDS is modeled by category morphisms that allow for expressing particular changes of states.

There are several interesting ways of defining a category (structure and graph), see e.g., [21–23]. Here, we give the following definition:

A category **C** consists of the following:

1. (Objects) A class of objects denoted $\mathbf{C}_{obj}$ (possibly $Obj(\mathbf{C})$, as well). Its elements are called the *objects*. Sometimes, it is customary to write $X \in \mathbf{C}$ instead of $X \in \mathbf{C}_{obj}$.
2. (Morphisms) For each pair of objects $X, Y \in \mathbf{C}$, a set $Hom(X, Y)$ is called the *hom-set* for the pair of objects (X, Y). The elements of hom-set $Hom(X, Y)$ are called *morphisms* (or equivalently *arrows*) from X to Y. If $f \in Hom(X, Y)$, we also write

$$f : X \to Y.$$

   The class of all morphisms of a category $\mathbf{C}$ is denoted by $\mathbf{C}_{morph}$ (possibly $Morph(\mathbf{C})$ or $Ar(\mathbf{C})$, as well).
3. (Identity morphism) For each object $X \in \mathbf{C}$, there is a morphism $\mathrm{id}_X \in Hom(X, X)$ (sometimes denoted also as 1_X), called the *identity morphism* for an object X, with the property that, if $f \in Hom(X, Y)$, then

$$\mathrm{id}_X \circ f = f \text{ and } f \circ \mathrm{id}_X = f.$$

4. (Composition) For two morphisms f, g, where $f \in Hom(X, Y)$ and $g \in Hom(Y, Z)$, there is a new morphism $g \circ f \in Hom(X, Z)$, called the composition of g with f. Moreover, a composition must meet the condition of associativity:

$$f \circ (g \circ h) = (f \circ g) \circ h,$$

   whenever the composition is defined.

2.2.1. Polynomial (Endo)Functors

A category provides an environment (or a context) where we can talk about composition such that properties such as associativity and identity hold. A connection of (category) environments is represented by structure-preserving mapping. The mapping that respects the categorical structure is called a *functor*.

A functor $F : \mathbf{C} \to \mathbf{D}$ between categories is a pair of functions (for both, we use the same symbol F):

- The object part of the functor:

$$F : \mathbf{C}_{obj} \to \mathbf{D}_{obj},$$

  which sends objects in $\mathbf{C}$ to objects in $\mathbf{D}$, and
- The morphism (arrow) part of the functor:

$$F : \mathbf{C}_{morph} \to \mathbf{D}_{morph},$$

  which maps morphisms in $\mathbf{C}$ to morphisms in $\mathbf{D}$.

These assignments are required to satisfy the following functoriality axioms:

- For any composable pair of morphisms in $\mathbf{C}$,

$$Fg \circ Ff = F(g \circ f);$$

- For each object $X \in \mathbf{C}$,

$$F(\mathrm{id}_X) = \mathrm{id}_{FX}.$$

Coalgebras play a special role as a polynomial endofunctor [24]. When we model a state space as a category, a polynomial endofunctor over this category characterizes (describes) the change of states. The notion "polynomial" comes from its shape: it is similar to polynomials because it can be constructed using the constants, products, coproducts, and exponentials as follows:

$$T(X) = \sum_{i,j=0}^{n} A_i \times X^{B_j},$$

where X stands for a state space, A_i are sets of observable values, and B_j are some fixed sets for $i, j \in \mathbb{N}$.

The signature selector determines the polynomial endofunctor [25]. Each polynomial endofunctor characterizes one specific type of system, e.g., streams, finite lists, deterministic automaton, non-deterministic or transition systems, etc. In our approach, we work with deterministic systems.

2.2.2. Coalgebra for a Polynomial Endofunctor

Let us consider a base category **C** and a polynomial endofunctor T over category **C**. Kurz formally defined a coalgebra for a polynomial endofunctor as an ordered pair in his work [26]

$$(X, \omega),$$

where

- X is the objects of category **C** and
- ω is a morphism of category **C** such as $\omega : X \to T(X)$.

Objects $X \in \mathbf{C}$ form the state space of a category, and the morphism $\omega \in \mathbf{C}$ is called the coalgebraic transition structure (or coalgebra dynamics); in the general case, it is an $n-$tuple of the selectors.

2.3. Logical Systems

Our logical model is based on linear logic and BDI logic; therefore, in this section, we present their necessary basic notions.

2.3.1. Linear Logic

Traditional logical systems such as propositional or predicate logic are not sufficient for a description of the exact behavior of complex program systems. Therefore, we chose a linear logic formulated in [27] as a main logical system. The linear logic has many advantages compared with other logical systems, such as a stronger expressive power achieved by introducing new logical connectives, which allows for expressing resource-based treatment of formulae, or a time-spatial theory called Ludics [28]. Each formula represents an action/reaction or an available/consumed resource.

We denote the elementary formulae by the Latin lowercase letters a, b, c, etc. and the formulae by the Greek lowercase letters $\varphi, \psi, \vartheta,$, etc. Below, we present the possible forms of the formulae in linear logic given by the Backus–Naur form:

$$\varphi ::= a \mid \mathbf{1} \mid \bot \mid \mathbf{0} \mid \top \mid \varphi \otimes \varphi \mid \varphi \,\&\, \varphi \mid \varphi \oplus \varphi \mid \varphi \parr \varphi \mid \varphi \multimap \varphi \mid \varphi^{\bot} \mid !\varphi \mid ?\varphi. \quad (1)$$

The elements of BNF (1) represent the following: a stands for an atomic formula; $\otimes$ represents parallelism; $\&/\oplus$ represent an outer/inner nondeterminism, respectively; $\parr$ represents an exclusive disjunction; $\multimap$ represents the dynamics when consuming resources; $()^{\bot}$ represents a consumed resource or a reaction; and $!/?$ represents an unlimited or a potentially unlimited resource, respectively. The particular constants $\mathbf{1}, \bot, \mathbf{0}$, and $\top$ stand for the neutral element of a logic for their corresponding operations: $\otimes, \parr, \&$, and $\oplus$, respectively. In [14], all connectives were described in detail.

2.3.2. BDI Logic

A BDI logic is a modal logical system. Originally introduced by Rao and Georgeff in their publication [29] as a model for reasoning about intelligent BDI agents, its origins are based on the BDI philosophical theory of Bratman [19]. Its syntax consists of logical operators from classical propositional logic, and it is extended with the three modal operators of belief, desires, and intentions. A BDI agent can be considered an autonomous entity that observes an environment through its "sensors", and it can act within in it.

The syntax of BDI logic language can be expressed by following BNF:

$$\varphi ::= p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi \mid \neg\varphi \mid BEL_a\varphi \mid DES_a\varphi \mid INT_a\varphi. \tag{2}$$

The modal formulae of belief, desire, and intention,

$$BEL_a\varphi, DES_a\varphi, INT_a\varphi, \tag{3}$$

express the beliefs, desires, and intentions of an BDI agent, respectively, and the relations between them, where a represents an agent's name.

3. Linear BDI Logic

In the first step, we start with the definition of an appropriate logical system for our model of an active intrusion detection system. We use linear logic [27] and a classical propositional BDI logical system [29]. We extend the language of linear logic with modal operators of *belief, desire,* and *intention* [18]. Such language increases the expressive power of original logical systems, which creates possibilities to express necessary processes, to define an optimal state of an environment, and to act upon events. In our approach, we define the semantics of linear BDI logic using the Kripke semantic method of possible worlds.

3.1. Syntax of Linear BDI Logic

We define our language as follows.

- Logical connectives are linear logic's connectives:

$$(.)^{\perp}, \&, \otimes, \multimap, \parr, \oplus. \tag{4}$$

- We extend the syntax with the modal operator of objective knowledge $K_i(\varphi)$.
- We leave the definition of modal operators BEL_i, DES_i, and INT_i as is.

Formally, the linear BDI logic syntax of our language can be expressed by following BNF:

$$\begin{aligned}\varphi ::= a \mid \mathbf{1} \mid \perp \mid \mathbf{0} \mid \top \mid \varphi \otimes \varphi \mid \varphi \,\&\, \varphi \mid \varphi \oplus \varphi \mid \varphi \parr \varphi \mid \varphi \multimap \varphi \mid \varphi^{\perp} \\ \mid K_i(\varphi) \mid BEL_i(\varphi) \mid DES_i(\varphi) \mid INT_i(\varphi),\end{aligned} \tag{5}$$

where the set of all formulae is denoted as $BDIform$. The logical connectives of our language have the following meanings:

- An informal description of the linear logic connectives is listed in Section 2.3.1.
- $K_i(\varphi)$ states that agent i knows φ;
- $BEL_i(\varphi)$ states that agent i believes in φ,
- $DES_i(\varphi)$ states that φ is desire of agent i,
- $INT_i(\varphi)$ states that φ is the intention of agent i.

3.2. Semantics of Linear BDI Logic

We defined the semantics of our logic using the Kripke model. Our definition for the Kripke model is an ordered tuple:

$$\mathbf{M_{BDI}} = (W, \leqq, \models, x), \tag{6}$$

where

- W is not an empty set of possible worlds $W = \{x_0, x_1, x_2, ...\}$;
- $\leqq$ is the binary relation of accessibility between worlds $\leqq W \times W$;
- $\models$ is the satisfaction relation, defined as follows:

$$\models: W \times BDIform, \tag{7}$$

where $x_n \models a$ means that an elementary formula a is valid in the world x_n; and
- x denotes a designated world $x \in W$.

Now, we define the satisfaction relation for each element of the production rule (5) for linear BDI logic.

$$
\begin{array}{lll}
\mathbf{M_{BDI}}, x \models a & \text{iff} & a \in A(x) \\
\mathbf{M_{BDI}}, x \models \mathbf{1} & & \\
\mathbf{M_{BDI}}, x \models \bot & & \\
\mathbf{M_{BDI}}, x \models \mathbf{0} & & \\
\mathbf{M_{BDI}}, x \models \top & & \\
\mathbf{M_{BDI}}, x \models \varphi^{\bot} & \text{iff} & \mathbf{M_{BDI}}, x \not\models \varphi \\
\mathbf{M_{BDI}}, x \models \varphi \otimes \psi & \text{iff} & \mathbf{M_{BDI}}, x \models \varphi \quad \text{and} \quad \mathbf{M_{BDI}}, x \models \psi \\
\mathbf{M_{BDI}}, x \models \varphi \oplus \psi & \text{iff} & \mathbf{M_{BDI}}, x \models \varphi \quad \text{and} \quad \mathbf{M_{BDI}}, x \models \psi \\
 & & \text{or} \quad \mathbf{M_{BDI}}, x \models \varphi \quad \text{or} \quad \mathbf{M_{BDI}}, x \models \psi \\
\mathbf{M_{BDI}}, x \models \varphi \parr \psi & \text{iff} & \mathbf{M_{BDI}}, x \models \varphi \quad \text{xor} \quad \mathbf{M_{BDI}}, x \models \psi \\
\mathbf{M_{BDI}}, x \models \varphi \,\&\, \psi & \text{iff} & \mathbf{M_{BDI}}, x \models \varphi \quad \text{or} \quad \mathbf{M_{BDI}}, x \models \psi \\
\mathbf{M_{BDI}}, x \models \varphi \multimap \psi & \text{iff} & (\forall x_n) x \leq x_n \quad \text{if} \quad \mathbf{M_{BDI}}, x_n \models \varphi \\
 & & \text{then } \mathbf{M_{BDI}}, x_n \models \psi \\
\mathbf{M_{BDI}}, x \models K_i(\varphi) & \text{iff} & (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi \\
\mathbf{M_{BDI}}, x \models BEL_i(\varphi) & \text{iff} & (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi \\
\mathbf{M_{BDI}}, x \models DES_i(\varphi) & \text{iff} & (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi \\
\mathbf{M_{BDI}}, x \models INT_i(\varphi) & \text{iff} & (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi
\end{array}
\tag{8}
$$

where $A(x)$ denotes a set of all atomic formulae that are valid in the world x in the model M.

4. Logical Model for Active IDS

The main goal of this paper is the formulation of a logical model for an active network-based intrusion detection system (IDS). We chose a system that detects network intrusions based on known patterns as a basis for our model. For the construction of a model, we use category theory and we define a new logical system based on BDI logic and linear logic. In the previous section, we formulated its semantics using the Kripke model of possible worlds.

The new logical system allows for the possibility to gain knowledge and beliefs about network intrusion, which is then confronted by the network security policy. If the detected event violates the system security policy, appropriate countermeasures/actions are selected to restore the system to the desired state. By formulating such a model, we also propose the extension of a standard IDS functionality using autonomous reactions to detect intrusions.

Our logical model for active IDS is formally depicted in Figure 2 *(the formulae are explained in the next sections)*. We divide it into two layers.

1. On the first layer, we define a coalgebra for a polynomial endofunctor that serves as a model of passive IDS. For a coalgebra, we need to consider the following:
   (a) Many-typed signatures for specifying network intrusions;
   (b) A category of packets as a state-space of IDS;
   (c) A polynomial endofunctor over that category;
   (d) Modeling the behavior of an IDS using a coalgebra for a polynomial endofunctor over a constructed category;
   (e) The description of a behavior of an IDS using the coalgebraic modal linear logic formula; and
   (f) Obtaining knowledge about the possible intrusions by the linear BDI.
2. The second layer is a model of an active IDS. It formally consists of the following parts:
   (a) Obtaining a belief of an intrusion by the linear BDI;
   (b) A definition of the security policy database—i.e., the desires; and
   (c) A definition of the countermeasures for a security policy's possible violations, i.e., the intentions.

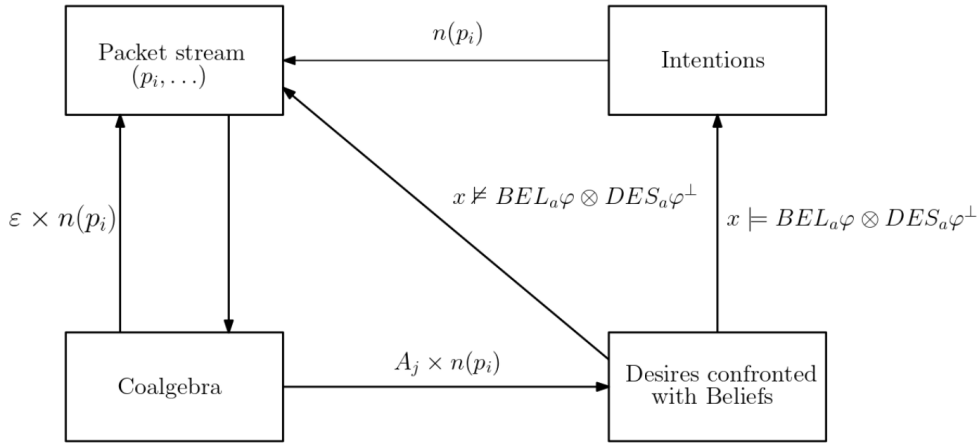


Figure 2. Logical model for active IDS.

4.1. Many-Typed Signatures of Network Intrusions

Network-based intrusion detection systems are divided into types based on a method of intrusion detection. In our work, we use an IDS that detects intrusions based on known patterns. Those patterns are called network intrusion signatures [30].

The first step of our logical model definition is a specification of network intrusion signatures as many-typed algebraic signatures. Based on that, we can extract specific symptoms of a particular intrusion and determine a coalgebra's selectors. The approach presented in this paper is demonstrated on a practical example of real network intrusions and reactions of a signature-based IDS. As a motivating example, we chose three specific network intrusions of a "Man-In-The-Middle" type:

- *Portscan* intrusion is a prerequisite attack technique. It sends each client a local area network request, intending to find some active ports. One can use known exploits to attack them [31].
- *ARP spoofing* uses a vulnerability of the ARP protocol to redirect network traffic [32].
- *SSLstrip* downgrades a connection from a secure HTTPS to an insecure HTTP [33].

The whole process of a specification is rather complex; therefore, here, we present only the results of our works published in [34,35]. Treated intrusions and their symptoms are listed in Table 1.

Table 1. Symptoms of intrusions.

NMAP	ARP Spoofing	SSLStrip
exter_net == any	exter_net == any	exter_net == any
home_net == any	home_net == any	home_net == any
home_port == 7	redir_host == any	redir_host == any
protocol == tcp	protocol == icmp	ttl == 1
flow == stateless	icode == 1	protocol == pim
tcp_flags == F,P,U,12	itype == 5	flow == stateless
	flow == stateless	

4.2. Category of Packets

In the second step of our approach, we constructed a category of packets $\mathcal{C}packets$ based on the definition coalgebra in Section 2.2.2, where

- *Objects* are packets $Packets = \{p_1, p_2, \ldots, \}$;
- The *morphism n* is specified as follows:

$$n : Packets \to Packets, \tag{9}$$

and defined as

$$n(p_i) = p_{i+1}, \tag{10}$$

which describes one transitional step between two packets in a stream, where $i \in \mathbb{N}$; and

- *Identity morphisms*: for each object $p_i \in Packets$, there exists its identity morphism id_{p_i}:

$$id_{p_i} : Packets \to Packets, \tag{11}$$

defined as follows:

$$id_{p_i}(p_i) = p_i. \tag{12}$$

We consider packets as an infinite stream, and we denote it as $Packets$, where

$$Packets = \{p_1, p_2, \dots\}, \tag{13}$$

and it is considered later as a state space of a coalgebra.

A model of the category $\mathcal{C}packets$ is depicted in Figure 3.

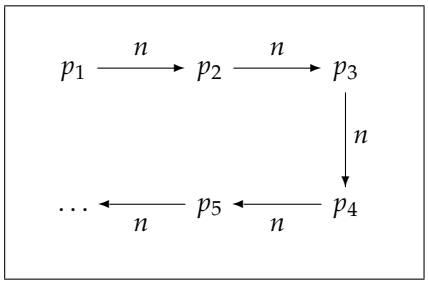


Figure 3. Category of an infinite stream of packets $\mathcal{C}packets$.

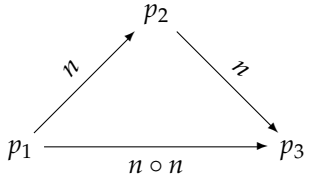
Theorem 1. *From the definition of category (Section 2.2), the category $\mathcal{C}$packets is a category if it holds composition and associative laws.*

Proof. We show that composition and associative laws are valid in the category $\mathcal{C}packets$.

- *Composition law*:
  Let $p_1, p_2, p_3 \in \mathcal{C}_{obj}$, and $n \in \mathcal{C}_{morf}$.
  - Let $n(p_1) = p_2$, and $n(p_2) = p_3$,
  - then the following holds:

$$\begin{array}{lcl} (n \circ n)(p_1) & = & p_3, \\ n(n(p_1)) & = & p_3, \\ (n \circ n)(p_1) & = & (n(n(p_1))). \end{array} \tag{14}$$

  This can be depicted by a commutative diagram as follows:



- *Associative law*:
  Let $p_1, p_2, p_3, p_4 \in \mathcal{C}_{obj}$, and $n \in \mathcal{C}_{morf}$.
  - Let $n(p_1) = p_2, n(p_2) = p_3$, and $n(p_3) = p_4$,

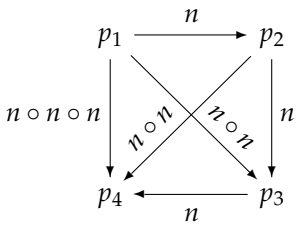
– then the following holds:

$$\begin{array}{lcl} n(n(n(p_1))) & = & p_4, \\ n(n \circ n)(p_1) & = & p_4, \\ (n \circ n)(n(p_1)) & = & p_4, \\ (n \circ n \circ n)(p_1) & = & p_4, \end{array} \tag{15}$$

Therefore,

$$\begin{array}{l} n(n(n(p_1))) = n(n \circ n)(p_1) = \\ = (n \circ n)(n(p_1)) = (n \circ n \circ n)(p_1). \end{array} \tag{16}$$

This can be depicted by a commutative diagram as follows:



□

4.3. Coalgebra Determined by the Polynomial Endofunctor over Category of $\mathcal{C}$ Packets

We formulate a polynomial endofunctor over a category $\mathcal{C}packets$ as follows:

$$T : \mathcal{C}packets \to \mathcal{C}packets. \tag{17}$$

We define the endofunctor as follows:

- For objects,

$$T(p_i) = \bigsqcup_{j=1}^{n} A_j \times n(p_i), and \tag{18}$$

- For morphisms,

$$T(p_i \mapsto p_{i+1}) = p_{i+1} \mapsto p_{i+2}, \tag{19}$$

where

- $i \in \mathbb{N}$;
- I denotes the number of known attacks, where $j \in I$;
- $A = \{\varepsilon, A_1, A_2, \ldots, A_n\}$ represents the class of specific known attacks and their characteristic symptoms, specified by signatures, where ε denotes a situation where no intrusion has been detected;
- $\bigsqcup A_j$ is a sum of the intrusions that occurred; and
- A function n is defined as follows $n : Packets \to T(Packets)$.
  Important symptoms of intrusions are in the form of equalities, defined as elementary formulae, as follows.
  - $A_1 = \{a_1, a_2, a_3, a_4, a_5, a_6\}$ represents the symptoms of a known attack NMAP;
  - $A_2 = \{b_1, b_2, b_3, b_4, b_5, b_6, b_7\}$ represents the symptoms of a known attack ARP Spoof; and
  - $A_3 = \{c_1, c_2, c_3, c_4, c_5, c_6\}$ represents the symptoms of a known attack SSLStrip.

Based on the definition of a polynomial endofunctor introduced, we coalgebraically model the IDS as a coalgebra for a polynomial endofunctor:

$$IDS = (Packets, n : Packets \to T(Packets)). \tag{20}$$

Now, we assume that a stream of packets consists of the following sequence:

$$(\ldots, p_i, p_{i+1}, p_{i+2}, p_{i+3}, \ldots), \tag{21}$$

which is monitored by the coalgebra defined:

$$\ldots, A_j \times T(p_i), A_k \times T(p_{i+1}), A_l \times T(p_{i+2}), A_m \times T(p_{i+3}), \ldots. \tag{22}$$

where A_j, A_k, A_l, A_{jm} represent specific intrusions from the class of known intrusions A.

The formulae of coalgebraic logic are used for logical reasoning over states of a dynamic system that is captured by the coalgebra for a polynomial endofunctor. Now, we specify a stream of packets as formulae of the modal language defined in Section 3.

The application of coalgebraic specification creates an infinite sequence of coalgebraic formulae:

$$\begin{aligned} &(\mathbf{1}) \\ &(p_1, \mathbf{1}) \\ &(p_1, (p_2, \mathbf{1})) \\ &(p_1, (p_2, (p_3, \mathbf{1}))) \\ &\ldots \\ &(p_1, (p_2, (p_3, (\ldots, (\ldots, \mathbf{1}) \ldots)))), \end{aligned} \tag{23}$$

where the first line $(\mathbf{1})$ denotes an empty sequence, considered the initial state of the system. The second line arises after the first application of coalgebraic specification. It specifies the initial (*starting*) packet. The next lines describe an iterative application of coalgebraic specification up to a possible infinite sequence. The last row corresponds with the following coalgebraic linear formula:

$$\bigotimes\{(p_1, (p_2, (p_3, (\ldots, (\ldots, \mathbf{1}) \ldots)))))\}. \tag{24}$$

4.4. From Knowledge to Belief

We extended the language of our logical system by the modal operators of knowledge ($K_a\varphi$) and belief ($BEL_a\varphi$).

$$\varphi ::= \quad \ldots \mid K_a\varphi \mid BEL_a\varphi. \tag{25}$$

For a definition of semantics, we use a Kripke model introduced in Section 3.2. Through this model, we demonstrate how knowledge and a belief about the system's intrusion can be acquired. The Kripke satisfaction relation is defined as follows:

$$\models: W \times BDIform. \tag{26}$$

In Jaakko Hintikka's publication [36] about semantics for ($K_a\varphi$) and ($BEL_a\varphi$), he stated that "whenever one knows something, one believes that one knows it" and that "whenever one beliefs something, one knows that one believes it". Based on that, we define the semantics for the operators of knowledge ($K_a\varphi$) and belief ($BEL_a\varphi$) as follows, where a is an agent and x is a designated world:

$$\begin{aligned} \mathbf{M_{BDI}}, x \models K_a(\varphi) \quad &\text{iff} \quad (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi, \\ \mathbf{M_{BDI}}, x \models BEL_a(\varphi) \quad &\text{iff} \quad (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi. \end{aligned} \tag{27}$$

The first line of the definition (27) states that a formula "an agent a knows that φ" has sense in a designated world x in a Kripke model $\mathbf{M_{BDI}}$. The second line states that a formula "an agent a believes that φ" has sense in a designated world x in a Kripke model $\mathbf{M_{BDI}}$.

From the statement above, we define a semantics for a knowledge operator as follows:

$$\text{if} \quad \mathbf{M_{BDI}}, x \models K_a(\varphi) \quad \textit{then} \quad \mathbf{M_{BDI}}, x \models BEL_a(\varphi), \tag{28}$$

which can be read as "if an agent a knows about φ, then it believes that φ in a designated world x, in a Kripke model $\mathbf{M_{BDI}}$".

Now, we assume that a stream of packets consists of the following sequence:

$$(\dots, p_1, p_2, p_3, p_4, \dots), \tag{29}$$

which is monitored by the defined coalgebra.

In the following example, the behavior of the system can be modeled in particular steps:

$$\begin{array}{llll} \cdots \mapsto & \varepsilon \times T(p_1) & \mapsto \\ \mapsto & A_1 \times T(p_2) & \mapsto \\ \mapsto & A_2 \times T(p_3) & \mapsto \\ \mapsto & A_3 \times T(p_4) & \mapsto \dots \end{array} \tag{30}$$

where ε denotes a situation, when no intrusion is detected in the treated packet. In the example above, the coalgebra "observes" a stream with the following results:

- p_1—no intrusion detected;
- p_2—an intrusion A_1 detected, which represents the *NMAP* intrusion;
- p_3—an intrusion A_2 detected, which represents the *ARP spoofing* intrusion; and
- p_4—an intrusion A_3 detected, which represents the *SSLStrip* intrusion.

Let following set be a set of atomic formulae:

$$\{a_1, a_2, a_3, a_4, a_5, a_6, \dots, b_1, b_2, b_3, b_4, b_5, b_6, b_7, \dots, c_1, c_2, c_3, c_4, c_5, \dots\}. \tag{31}$$

Each atomic formula denotes one symptom of possible intrusions. Our motivation example deals with three intrusions, with their specific symptoms based on the detection signatures. Treated intrusions and their symptoms are depicted in Table 2:

Table 2. Atomic formulae as intrusion symptoms.

A_1 : NMAP s	A_2 : ARP Spoofing	A_3 : SSLStrip
a_1 : exter_net = any	b_1 : exter_net = any	c_1 : exter_net = any
a_2 : home_net = any	b_2 : home_net = any	c_2 : home_net = any
a_3 : home_port = 7	b_3 : redir_host = any	c_3 : redir_host = any
a_4 : protocol = tcp	b_4 : protocol = icmp	c_4 : ttl = 1
a_5 : flow = stateless	b_5 : icode = 1	c_5 : protocol = pim
a_6 : tcp_flags = F,P,U,12	b_6 : itype = 5	c_6 : flow = stateless
	b_7 : flow = stateless	

An agent a gains knowledge about a specific intrusion in a Kripke world if all symptoms are valid in this world. In our example:

- The intrusion A_1 can be detected if, in some possible worlds x_n, all symptoms $(a_1 - a_6)$ are valid. This means that 6 symptoms create 64 possible worlds $(x_1 - x_{64})$, where only 1 is designated as the world

$$\mathbf{M_{BDI}}, x_n \models A_1 \quad \text{iff} \quad \mathbf{M_{BDI}}, x_n \models a_1 \otimes a_2 \otimes a_3 \otimes a_4 \otimes a_5 \otimes a_6. \tag{32}$$

Let the designated world be x_1. Then,

$$\mathbf{M_{BDI}}, x_1 \models K_a(\bigotimes_{s \in A_1} a_s) \quad \text{iff} \quad \mathbf{M_{BDI}}, x_1 \models K_a(A_1). \tag{33}$$

- The intrusion A_2 can be detected if, in some designated world x_n, all symptoms ($b_1 - b_7$) are valid. This means that 7 symptoms create 128 possible worlds ($x_{65} - x_{192}$), where only 1 is the designated world

$$\mathbf{M_{BDI}}, x_n \models A_2 \quad \text{iff} \quad \mathbf{M_{BDI}}, x_n \models b_1 \otimes b_2 \otimes b_3 \otimes b_4 \otimes b_5 \otimes b_6 \otimes b_7. \tag{34}$$

Let the designated world be x_{65}. Then,

$$\mathbf{M_{BDI}}, x_{65} \models K_a(\bigotimes_{s \in A_2} b_s) \quad \text{iff} \quad \mathbf{M_{BDI}}, x_{65} \models K_a(A_2). \tag{35}$$

- The intrusion A_3 can be detected if, in some designated world x_n, all symptoms ($c_1 - c_6$) are valid. This means that 6 symptoms create 64 possible worlds ($x_{193} - x_{256}$), where only 1 is the designated world

$$\mathbf{M_{BDI}}, x_n \models A_3 \quad \text{iff} \quad \mathbf{M_{BDI}}, x_n \models c_1 \otimes c_2 \otimes c_3 \otimes c_4 \otimes c_5 \otimes c_6. \tag{36}$$

Let the designated world be x_{193}. Then

$$\mathbf{M_{BDI}}, x_{193} \models K_a(\bigotimes_{s \in A_3} c_s) \quad \text{iff} \quad \mathbf{M_{BDI}}, x_{193} \models K_a(A_3). \tag{37}$$

From the definition of a belief operator's semantics (28), the agent *a* "*believes*" that an intrusion happened in a designated world if it "*knows*" that an intrusion happened in a designated world.

$$\begin{array}{llll} \text{if} & \mathbf{M_{BDI}}, x_1 \models K_a(A_1) & \textit{then} & \mathbf{M_{BDI}}, x_1 \models BEL_a(A_1), \\ \text{if} & \mathbf{M_{BDI}}, x_{65} \models K_a(A_2) & \textit{then} & \mathbf{M_{BDI}}, x_{65} \models BEL_a(A_2), \\ \text{if} & \mathbf{M_{BDI}}, x_{193} \models K_a(A_3) & \textit{then} & \mathbf{M_{BDI}}, x_{193} \models BEL_a(A_3). \end{array} \tag{38}$$

The agent *a* now obtains beliefs about the three intrusions A_1, A_2, and A_3. The next step is to create appropriate countermeasures based on the network security policies.

4.5. Desires

The layer *desires* represents the security policy of a network's organization. Network security policies are mostly defined informally by natural language sentences. This creates certain ambiguities within a policy. In our approach, we define it as a sequence of logical formulae, i.e., *desires*, that has to be valid. For that, we extended the language of our logical system using the modal operators of desire ($DES_a\varphi$)

$$\varphi ::= \ \ldots \mid DES_a\varphi, \tag{39}$$

and we defined its Kripke satisfaction relation as follows:

$$\mathbf{M_{BDI}}, x \models DES_a(\varphi) \quad \text{iff} \quad (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi, \tag{40}$$

Once an agent obtains a belief about some malicious activity, it is checked with the agent's desires for the monitored system; after that, appropriate countermeasures are taken. Therefore, we define a database of an agent's desires, which represents estimated state of the system. In the case of IDS, an agent's desires are opposite to the detection of intrusions. That means that an agent's desires are defined as follows:

$$\text{if} \quad \mathbf{M_{BDI}}, x \models BEL_a A_I \quad \textit{then} \quad \mathbf{M_{BDI}}, x \models DES_a(A_I)^{\perp} \tag{41}$$

Security policies are defined by an individual organization; therefore, one can define other desires or choose which detected intrusions are not against a policy. In our case, we defined a database of desires (41) simply for our motivation example.

For our motivation example, we define the following desires:

- $DES_a A_1^{\perp}$: an agent a does not desire an intrusion A_1, which represents the *NMAP* intrusion;
- $DES_a A_2^{\perp}$: an agent a does not desire an intrusion A_2, which represents the *ARP spoofing* intrusion; and
- $DES_a A_3^{\perp}$: an agent a does not desire an intrusion A_3, which represents the *SSLStrip* intrusion.

Therefore, consider that the three desires of a security policy were broken.

4.6. Intentions

In the case of IDSs, intentions represent the countermeasures (i.e., an agent's plans) that should be taken into account in order to bring about/restore the desired state of the system. Each plan is constructed to restore this desired state (i.e., to prohibit and prevent future possible breaches in a security policy). An agent starts executing an intention only if it believes an intrusion occurred that is in contradiction with its desires. On the other hand, if the belief about an intrusion is not against a security policy, then no action is undertaken. For that, we extended the language of our logical system using the modal operators of intention ($INT_a\varphi$).

$$\varphi ::= \ \ldots \mid INT_a\varphi, \tag{42}$$

and we defined its Kripke satisfaction relation as follows:

$$\mathbf{M_{BDI}}, x \models INT_a(\varphi) \quad \text{iff} \quad (\forall x_n) x \leq x_n : \mathbf{M_{BDI}}, x \models \varphi, \tag{43}$$

This process is described by the following formulae.

$$x \models (BEL_a\varphi \otimes DES_a\varphi^{\perp}) \multimap INT_a\varphi_I, \tag{44}$$

where Equation (44) represents a situation where the desires of an agent a are in contradiction with the agent's beliefs. Therefore, it implies that an appropriate countermeasure has to be undertaken to restore the desired state. This means that, after a plan is executed, the agent stops believing that an intrusion has occurred.

$$\mathbf{M_{BDI}}, x \models INT_a\varphi_I \multimap BEL_a\varphi^{\perp}. \tag{45}$$

After the defined intention takes place (φ_I), an agent a loses their belief of an intrusion (φ).

The formula

$$\mathbf{M_{BDI}}, x \not\models (BEL_a\varphi \otimes DES_a\varphi^{\perp}) \multimap \perp, \tag{46}$$

represents a situation where an agent's beliefs are not in conflict with its desires and where, therefore, no intentions are acted upon.

The intrusions are divided into different types, e.g., Man-In-The-Middle, Denial of Service, etc. Usually, it is necessary to configure the system only against one type to prevent the whole group of intrusions.

We define an intention $INT_a\varphi_{I_j}$ as follows:

- a is the atomic formulae: i.e., actions $a_1, a_2, \ldots, a_n$; they represent specific actions that should be taken in order to deal with an intrusion. Some of the elementary formulae that are defined for an individual intrusion can be used as the parameters of an intention's action, e.g., the IP address of an intruder.
- φ_{I_j} represents a specific intention, i.e., a plan constructed from elementary action for each type of intrusion I_j or one specific intrusion.

For our motivation example, we define following intentions.

- $I_{A_{MITM}}$ for intrusions A_1, A_2, and A_3, with following atomic formulae:
  1. a_1—Add an attacker's IP address to the list of blocked IP addresses.
  2. a_2—Create an appropriate log about an intrusion.

3. a_3—Send an email to the system administrator.

In this motivation example, an agent a deals with their intrusions, which is of the MITM type. After the detection of an intrusion, agent a obtains a belief that an intrusion occurred and checks if this situation is against the network's security policy defined by the database of desires. After that, an agent creates a plan with countermeasures to prevent future occurrences of intrusions from the same intruder.

5. Conclusions

In this paper, we presented a new approach that extends the current functionalities of intrusion detection systems using active reactions to detect intrusions. Our main result is an exact formal model based on category theory and logical systems. The results of this paper are based on our long-term work at formally modeling complex systems. Here, we present a new formal model of such a IDS, which in general, can improve network security.

The passive part of our active model of an IDS is the detection of an intrusion. We used a real implementation of a network IDS and its network intrusion detection signatures for its definition. Based on that part of our model of a passive IDS, we can obtain logical knowledge about an intrusion. This serves as an "input sensor" for the BDI part of our model. At this point of our model, we formulated a new approach to autonomous IDSs using the belief–desire–intention philosophy.

Following an exact formal design, one can also increase the reliability and correctness of a real IDS implementation. Such a proposed model with a concrete security policy can serve as a design model for the implementation of a concrete IDS in a real environment.

Such a model based on coalgebras and categories is clearly formulated without the loss of exactness. This approach can also help make formal methods more attractive in other areas of information and communication technologies. It can also contribute to education, where it is possible to show the possible practical use of logic and other formal methods.

Author Contributions: Conceptualization, J.P. and V.N.; methodology, J.P.; validation, J.P., W.S., and Z.B.; formal analysis, J.P., V.N., and Z.B.; investigation, J.P. and W.S.; resources, J.P. and V.N.; data curation, J.P. and W.S.; writing—original draft preparation, J.P. and V.N.; writing—review and editing, J.P., V.N. and W.S.; visualization, J.P.; supervision, V.N.; project administration, W.S.; funding acquisition, Z.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Cultural and Educational Grant Agency (Kultúrna a edukačná grantová agentúra MŠVVaŠ SR) grant No. KEGA 011TUKE-4/2020: "A development of the new semantic technologies in educating of young IT experts" and the Operational Programme Integrated Infrastructure within the project co-financed by the European Regional Development Fund code ITMS2014+: 313011V422: "Intelligent systems for UAV real-time operation and data processing", and the APC was funded by the Faculty of Electrical Engineering and Informatics at the Technical University of Košice under contract No. FEI-2021-76: "A Modern Interpreter of Predicate Linear Logic Formulas".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [30].

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liao, H.J.; Lin, C.H.R.; Lin, Y.C.; Tung, K.Y. Intrusion detection system: A comprehensive review. *J. Netw. Comput. Appl.* **2013**, *36*, 16–24.
2. Axelsson, S. *Intrusion Detection Systems: A Survey and Taxonomy*; Technical report; Department of Computer Engineering, Chalmers University of Technology: Goteborg, Sweden, 2000.
3. Perháč, J.; Mihályi, D.; Novitzká, V. Design of verifiable model of program systems' complex security using coalgebras and coalgebraic logics. In *Electrical Engineering and Informatics 7 : Proceedings of the Faculty of Electrical Engineering and Informatics of the Technical University of Košice*; Technical University of Košice: Košice, Slovakia, 2016; pp. 120–124.

4. Mihályi, D.; Novitzká, V. Towards to the Knowledge in Coalgebraic model IDS. *Comput. Inform.* **2014**, *33*, 61–78.
5. Mazal, Z.; Koci, R.; Janousek, V.; Zboril, F., Jr. Modelling intelligent agents for autonomic computing in the PNagent framework. *Int. J. Auton. Comput.* **2009**, *1*, 121–139.
6. Nunes, I.; Luck, M. Softgoal-based plan selection in model-driven bdi agents. In Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems. International Foundation for Autonomous Agents and Multiagent Systems, Paris, France, 5–9 May 2014; pp. 749–756.
7. Braubach, L.; Pokahr, A.; Lamersdorf, W. Jadex: A BDI-agent system combining middleware and reasoning. In *Software Agent-Based Applications, Platforms and Development Kits*; Springer: Basel, Switzerland, 2005; pp. 143–168.
8. Dastani, M.; Steunebrink, B. Modularity in bdi-based multi-agent programming languages. In Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, Milan, Italy, 15–18 September 2009; Volume 2, pp. 581–584.
9. Sun, Q.; Yu, W.; Kochurov, N.; Hao, Q.; Hu, F. A multi-agent-based intelligent sensor and actuator network design for smart house and home automation. *J. Sens. Actuator Netw.* **2013**, *2*, 557–588.
10. Sangulagi, P.; Sutagundar, A.; Manvi, S.; Bennur, V.S. BDI Agents for Information Fusion in Wireless Sensor Networks. *Int. J. Adv. Res. Comput. Eng. Technol. IJARCET* **2012**, *1*, 7.
11. Boudaoud, K.; Guessoum, Z.; McCathieNevile, C.; Dubois, P. *Policy-Based Security Management Using a Multi-Agent System*; Workshop HPOVUA: Berlin, Germany, 2001.
12. Lin, J.; Blythe, J.; Clark, S.; Davarpanah, N.; Hughston, R.; Zyda, M. Unbelievable Agents for Large Scale Security Simulation. Association for the Advancement of Artificial Intelligence. 2010. Available online: http://www.mikezyda.com/resources/pubs/SecArt10.pdf (accessed on 30 July 2021).
13. Kripke, S.A. Semantical analysis of intuitionistic logic I. *Stud. Log. Found. Math.* **1965**, *40*, 92–130.
14. Girard, J.Y. *The Blind Spot*; Lectures on Proof-Theory; Institut de Mathématiques de Luminy: Marseille, France, 2011.
15. Moss, L.S. Coalgebraic logic. *Ann. Pure Appl. Log.* **1999**, *96*, 277–317.
16. Singh, M.P.; Rao, A.S.; Georgeff, M.P. Formal methods in DAI: Logic-based representation and reasoning. In *Multiagent Systems*; MIT Press: Cambridge, MA, USA 1999; pp. 331–376.
17. Perháč, J.; Mihályi, D.; Mat'aš, L. Elimination of network intrusions via a resource oriented BDI architecture. *Open Comput. Sci.* **2018**, *8*, 173–181.
18. Perháč, J.; Mihalyi, D.; Mat'aš, L. Resource oriented BDI architecture for IDS. In Proceedings of the 2017 IEEE 14th International Scientific Conference on Informatics, Poprad, Slovakia, 14–16 November 2017; pp. 293–298.
19. Bratman, M.E.; Israel, D.J.; Pollack, M.E. Plans and resource-bounded practical reasoning. *Comput. Intell.* **1988**, *4*, 349–355.
20. Ehrig, H.; Mahr, B. *Fundamentals of Algebraic Specification 1: Equations and Initial Semantics*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012; Volume 6.
21. Barr, M.; Wells, C. *Category Theory for Computing Science*; Prentice Hall International (UK) Ltd.: Hertfordshire, UK, 1998.
22. Brandenburg, M. *Einführung in die Kategorientheorie: Mit Ausführlichen Erklärungen und Zahlreichen Beispielen*; Springer: Berlin/Heidelberg, Germany, 2016.
23. Walters, R.F.C. *Categories and Computer Science*; Cambridge Computer Science Texts; Cambridge University Press: Cambridge, UK, 1992.
24. Kock, J. Notes on Polynomial Functors. Manuscript, Version. 2009. Available online: https://mat.uab.cat/~kock/cat/polynomial.pdf (accessed on 30 July 2021).
25. Pierce, B.C.; Pierce, B.C.; Benjamin, I. *Basic Category Theory for Computer Scientists*; MIT Press: Cambridge, MA, USA 1991.
26. Kurz, A. *Logics for Coalgebras and Applications to Computer Science*; BoD–Books on Demand: Nordstedt, Germany, 2001.
27. Girard, J.Y. Linear logic. *Theor. Comput. Sci.* **1987**, *50*, 1–101.
28. Girard, J.Y. Locus Solum: From the rules of logic to the logic of rules. *Math. Struct. Comput. Sci.* **2001**, *11*, 301–506.
29. Rao, A.S.; Georgeff, M.P. Modeling Rational Agents within a BDI-Architecture. *KR* **1991**, *91*, 473–484.
30. Roesch, M. SNORT Users Manual. Available online: https://www.snort.org/ (accessed on 18 October 2018).
31. Lyon, G.F. *Nmap Network Scanning: The official Nmap Project Guide to Network Discovery and Security Scanning*; Insecure.Com LCC: Sunnyvale, CA, USA, 2009.
32. Whalen, S. An Introduction to Arp Spoofing. 2001. Available online: http://index-of.es/Misc/pdf/arp_spoofing_slides.pdf (accessed on 30 July 2021).
33. Marlinspike, M. More Tricks for Defeating SSL in Practice. Black Hat USA. 2009. Available online: http://2015.hack.lu/archive/2009/moxie-marlinspike-some_tricks_for_defeating_ssl_in_practice.pdf (accessed on 30 July 2021).
34. Perháč, J. Network Intrusion Detection Signatures Specified as Coalgebraic Many-typed Signatures. 2018. Available online: http://poseidon2.feld.cvut.cz/conf/poster/proceedings/Poster_2018/Section_IC/IC_058_Perhac.pdf (accessed on 30 July 2021).
35. Perháč, J.; Mihályi, D. Coalgebraic specification of network intrusion signatures. *Stud. Univ. Babes-Bolyai Inform.* **2016**, *61*, 83–94.
36. Jaakko, H.; Vincent, F.; Hendricks, J.S. Knowledge and Belief: An Introduction to the Logic of the Two Notions. *Philos. Rev.* **1965**, *74*, 381–384.