BEAST: Behavior as a Service for Trust Management in IoT Devices

Brennan Huber[a], Farah Kandah[a,1], Anthony Skjellum[b]

[a]*Computer Science and Software Engineering, Auburn University, Auburn AL 36849 USA*
[b]*SimCenter, University of Tennessee at Chattanooga, Chattanooga TN 37403 USA*

Abstract

As the internet becomes intertwined into every aspect of human life, the security of the Internet of Things (IoT) is also becoming increasingly critical. IoT devices are becoming the primary data source for a variety of smart-city applications, where critical decisions are based on this collected data. If malicious actors gain control of and/or tamper with the data being transmitted, the integrity of an entire smart city will be compromised. However, through monitoring IoT devices' behavior, anomalies can be detected and isolated to avoid any negative impact on decision-making. This behavioral monitoring process will complement traditional trust management approaches, since more accurate trust values can be calculated without the need to rely on a majority consensus. In this work, we present a BEhavior-As-a-Service for Trust management (BEAST) that implements a deep learning-based behavioral model to accurately classify IoT devices' interactions in the system. Through the implementation of the Elo rating system, these classifications will be presented as a vector of behaviors per device, which dynamically reflects each device's trust in the system. This work presents an analysis of our methodology as well as a threat model. Using simulations, a real-world use case is presented showing the interactions between IoT-based devices. Our results show that our BEAST model is able to dynamically evaluate each IoT device's trust, as well as capture and mitigate multiple threats targeting the trust in the system.

Keywords: Internet of Things, Trust Management, Security, Deep Learning, Behavioral Analysis, Smart City

1. Introduction

The rapid growth of connected devices comprising the Internet of Things (IoT) is transforming traditional elements of city life into next-generation intelligent smart cities where decisions are based on data being collected by IoT devices in real-time. As IoT devices deployment becomes more popular in different smart-city setups, these devices will make their way into every aspect of human life. This increase in IoT devices will also provide attackers with a new avenue to compromise critical infrastructure [1, 2, 3, 4, 5].

Among the key components contributing to smart-city initiatives is the intelligent transportation system (ITS), including connected vehicles. Connected vehicle technologies enable vehicles to communicate with their peers (V2V), roadside units (RSUs) (V2R), and other infrastructure (V2I) to share vital transportation information such as current road conditions, congested traffic, and vehicular collisions [6, 7]. If every vehicle had the necessary sensors and the ability to produce warning messages, the National Highway Traffic Safety Administration (NHTSA) predicts that the vehicles would prevent between 400,000-600,000 crashes, prevent between 190,000-270,000 injuries, and save close to 1,000 lives each year [8]. These technologies could prevent nearly 80% of all non-alcohol-related incidents [8], making it imperative to protect the data transmitted between these vehicles.

Although monitoring of malicious IoT devices is only in its infancy, the number of attackers is expected to drastically rise equally as the popularity of IoT rises [3, 4, 5]. IoT is the heart of smart cities, and because smart cities are constantly evolving with new trends and technology, it is vital to rethink cybersecurity designs to cope with this dynamic and evolving environment. Traditional cybersecurity approaches, such as asymmetric encryption through public key infrastructure, are not suitable in such setups considering IoT devices' limitations (computational power, battery, etc.) and the real-time requirements of smart cities [9, 10, 11]. For instance, ITS has a high mobility characteristic that brings additional challenges such as frequently changing topology, evaluation of the credibility of the devices and the messages, and inability to apply strong cryptographic measures to cope with the real-time requirement of ITS [12]. In such a collaborative environment, it is vital to provide a way to represent the bonding level among these devices as they share messages in the system. This can be achieved by managing the trust between devices in the network based on the accuracy of shared messages, which can relay a metric that represents a bonding level to other devices in the network. This approach was proposed to deal with the aforementioned challenges [13, 14, 15, 16, 17, 18].

Trust management has proved to be effective against the challenges that smart cities and ITS suffer from because trust management relies on a peer-driven network where devices themselves are responsible for detecting when another vehicle or device is

*Corresponding author
Email addresses: bmh0093@auburn.edu (Brennan Huber), farah-kandah@auburn.edu (Farah Kandah), tony-skjellum@utc.edu (Anthony Skjellum)

behaving maliciously. Trust itself is the evaluation of different aspects of a device that together represents the confidence, faith, and level of expectation of the device. Forming trust among devices in the network implicitly reflects the confidence level of the data being shared by such devices. For example, receiving data from a highly trusted device will provide a measure of assurance that the data is expected to be accurate.

However, traditional trust management systems rely on a majority consensus to determine accuracy. Additional challenges arise, such as when attackers work together to inject the network with malicious data to compromise devices by distorting the trust values of other devices in the network, and/or raising/lowering the trust values of a specific device to spoof certain changes in the network [19, 20, 21, 22]. Also, devices can manipulate the system by initially sending out accurate data to build a high trust level, then impact the system by sending misleading information that other devices will accept with confidence as it came from a highly trusted device [23]. Clearly, relying on peer-determined consensus is not enough to mitigate threats, making it both critical and urgent to design and implement a more robust trust management approach that is capable of sustaining the smart cities' requirements and at the same time monitoring devices' activities in real-time.

In this work, we present a dynamic and adaptable BEhavior-As-a-Service for Trust management (BEAST) in IoT Devices [1] to support the real-time requirements of IoT applications. Our approach builds a framework to generate a multidimensional behavioral model (behavioral vector), taking into account applications requirement along with the reported messages to identify different behaviors in concert with different incidents and reports generated by IoT devices. This behavioral vector will later be mapped to a trust value indicating the trustworthiness of the devices in the network. This provides a proactive dynamic layer of defense that enables the system to vet all devices that are participating and therefore be able to block devices trying to inject anomalous data that could manipulate system decisions.

This framework enables designers of highly automated systems, such as smart cities, to make decisions in real-time with high confidence in the data being received.

The remainder of the paper is organized as follows: We discuss related work in Section 2, followed by our motivations and contributions in Section 3, and the problem statement in Section 4. The threat model is presented and discussed in Section 5. Our BEhavior-As-a-Service for Trust management (BEAST) approach is described in Section 6. Next, Section 7 describes the system evaluation and experimental results. Finally, we conclude the paper in Section 8.

2. Related Work

This section provides a brief overview of Trust management concept and the trends in research that have led to the innovation of our work presented here.

[1] This work is an extension of our previous work: "Behavioral Model based Trust Management design for IoT at Scale(BLAST)" [18]

The concept of trust was ingrained in the field of sociology where it was defined as *"the extent to which one party is willing to participate in a given action with a given partner, considering the risks and incentives involved,"* as provided and adapted by [24, 25, 26]. While, trust management is the approach of evaluating trust such that it can be applied to computer systems enabling reputations to be built between peers to allow highly automated systems to make decisions with high levels of confidence — adapted by [12, 13, 14, 15].

Trust management is applied to systems that require high confidence in their decisions but also require these decisions to be made quickly. Most trust management algorithms explore this trade-off to optimize the confidence in data and timeliness of a decision [27]. This is the reason that trust management is often applied to critical infrastructure that has real-time requirements such as smart cities [28], connected vehicles [11, 29], smart grids [30], among others [31, 32, 33]. Additionally, trust has been shown to mitigate several security threats such as Distributed Denial of Service (DDoS), Sybil, replayed messages, and impersonation attacks [12]. However, protecting against such attacks is not enough because malicious actors can also seek to attack the trust management scheme itself by sending falsified data [34, 35], bad-mouthing peers [36], breakout fraud [37], or collusion attacks [38]. These attacks are significantly more dangerous to trust management algorithms since they have the ability to impact the integrity of the system by means of inflating the malicious actors' trust to a point where it becomes trustworthy even though it is behaving maliciously.

For trust management algorithms to work, it is necessary to ensure the trust-value-calculation component is calibrated such that, upon calculation, the device's trust value accurately reflects its overall trust in the network. Generally, trust values are a single percentage value that represents the likelihood that a device is telling the truth (i.e., a device with 100% trust will always tell the truth) [12, 28].

Several methodologies for calculating trust have been proposed recently. The first is algorithmic, where a device's trust value is incremented by a specified amount when a message is evaluated to be trustworthy and, conversely, decremented the specified amount upon a negative evaluation [14, 38]. While this trivial approach has been shown to reflect the relationships between devices, it is often insufficient since it enables attackers to game the system (e.g., they could attempt to build a high trust value and begin behaving maliciously without suffering severe consequences [39]).

Other trust management approaches have designed extensive trust-value-calculation components that integrate several factors [40, 41]. These factors can include a device's direct and indirect trust [42, 43] or other factors such as the participation rate, how long the device has been in the system, and/or the criticality of the messages sent [39].

One multi-factor approach is to calculate trust based on how a device is behaving in the system [39]. By applying weights to the specific factors, it allows additional flexibility in calculating trust such that specific factors can be more favored in the overall trust calculation. Additionally, obtaining a true trust value allows for the application with the use of the trust management to have

the ability to decide a threshold for determining the level of trustworthiness for allowing devices to participate in the system or be removed instantly.

Fuzzy logic has been used in these multi-factor trust-value-calculation components, not necessarily to calculate a single value but instead classify a device trust as either trustworthy or untrustworthy [44, 45]. While it has been shown that combining several factors to form a single classification is an effective means of trust management [46, 47], not having a true trust value eliminates the flexibility necessary to support real-time-based applications' setups, which enables highlighting devices that fell below a specific trustworthiness level indicating that these devices are not well fit to be part of the system and therefore need to be quarantined or removed from the system.

As previously mentioned, the increase or decrease in the trust of a device is based on the fact as to whether a message is evaluated to be legitimate or malicious [12, 28]. However, in the literature, these messages are vastly different and mainly unique to the application to which they are being applied, regardless, the general classification for trust management evaluating message techniques is broken into three categories: entity-oriented, data-oriented, and hybrid [11, 12]. Entity-oriented trust classification is based on the principle of monitoring the device itself, namely monitoring the metadata of the messages (IP address, MAC address, message size, etc) while data-oriented trust is focused on monitoring the specific data contained within the payload of a message [12]. While entity-oriented trust models have shown to be effective at mitigating attacks such as DDoS, reply, and Sybil attacks, such models are not enough because a new generation of threats has emerged aiming to mislead and manipulating trust management algorithms themselves [39]; as such it is necessary to design a data-oriented trust approach.

Recent work in trust management has applied machine learning to assist with increasing the accuracy of message evaluation as well as decreasing the time to process a message [48, 49]. By using anomaly detection or classification techniques, trust management algorithms are able to determine whether a message is trustworthy [37, 50]. However, most anomaly detection trust methods are focused on an entity-oriented approach; yet, it is critical to shift the focus to a data-oriented approach to ensure trust management systems are secure against all threats.

Other IoT data verification/trustworthiness mechanisms have been proposed including real-time software visualizations tools to monitor accuracy[51], data validation and error cleaning scheme to remove erroneous data [52], and data fusion technique to verify calibration of sensors to ensure accurate data is collected [48]. However, these trust management approaches offer a multifaceted approach that not only determines the trustworthiness of the data but that of the device itself. Because of these additional features, we are convinced that trust management is overall a better approach for verifying data trustworthiness.

3. Motivations and Contributions

Our motivations behind this work, and after studying the literature, are characterized by the following observations: ***(i)*** Traditional trust management systems are primarily focused on entity-oriented data and classifying devices as trustworthy or untrustworthy. Other approaches that consider a data-oriented approach heavily rely on a consensus of the data to determine on whether the message is trustworthy. Both of these types of trust management are prone to threats that directly target trust evaluation such as breakout fraud and colluding attacks. ***(ii)*** Current trust management implementations [11, 31, 32, 33, 53] also do not meet the real-time requirement of highly automated systems such as smart cities. It often takes several instances of malicious behavior before the application can safely and reliably prevent harm from devices that suddenly begin acting maliciously. Thus, it is vital to find a new implementation that can better prevent threats manipulating the trust management approach, while also maintaining the ability to process and mitigate the effects of malicious actors in real-time. ***(iii)*** The collaborative IoT environment increases the potential for cybersecurity threats that affect the way these devices communicate with one another and the system's decisions being made, which could potentially be life threatening human. ***(iv)*** The behavior of the IoT devices could change over time and could negatively affect the collaborative decisions in the area, thus skewing the decisions and affecting the outcome of the system while also impacting people's wellbeing. ***(v)*** The real-time requirement of IoT application is vital to the decision-making process, which makes it even more important to accurately detect and predict outcomes based on reported data from devices within the applications network [54, 55].

To that end, we summarize our contribution as follows:

(1) Construct a behavioral model:
This model will focus on monitoring each IoT device's reported data to form a localized behavioral model based on geographical location and standard behavior of devices in that area. This approach involves monitoring each device's data to gather an understanding of a specific geographical location and the typical behavior of the devices that participate there. Monitoring IoT device's data is vital to the real-time nature of the system because to detect and mitigate threats in real-time, the system needs to have the most recent data available. Collecting data must be done for a relatively small geographical area so that the system's ability to perform operations and respond in real-time is not affected. This behavioral model forms the foundation of the remainder of the work and can be expanded upon by creating many different behavioral models that each correspond to its own geographical area.

(2) Develop a behavioral pattern identification process:
Through the consistent monitoring of devices' behavior and using deep learning neural networks, it will be possible to detect, classify, and mitigate any abnormal behavior in the system in real-time. Using the behavioral model and the continued monitoring of devices' data, the system will go through a behavioral pattern identification process in which a device's current behavior will be compared against the known behavioral model. Any device that does not reasonably match the standard or expected behavior is determined to be an anomaly and will have its behavioral value adjusted accordingly. The analysis will provide critical insight into the detection and, more importantly, enable accurate mitigation of threats in the system.

(3) Formulate an Elo-based behavioral vector for trust man-

Table 1: Threats related to the impacted surface

Impact surface	Breakout fraud	On-Off attack	Colluding attack	Illusion-based attacks
Misleading impact	X	-	X	X
Response accuracy impact	X	X	X	X
Immediate system impact	X	-	X	X
Prolonged system impact	-	X	X	-

agement:
This component will focus on evaluating the behavior of a device by using the behavioral pattern identification process to determine a device's trust value. To further aid in determining the perceived trustworthiness of a device, an Elo-based formula will be used to derive a distinct behavioral value that correlates to a given device's likelihood of following the established pattern. Devices will have the ability to transmit messages with numerous ranks each having a unique meaning, so each device will have a behavioral value associated with each rank forming a behavioral vector. This behavioral vector can then be used in future implementations of trust management systems to more accurately formulate a device's overall trustworthiness.

4. Problem Statement

In this section, we define the terms being used throughout the discussion and formulate the problem statement.

Definition 4.1 (Trust/trustworthiness)**.** In this work, trust is a numerical value that distinctly represents a specific device's likelihood of providing accurate information on the environment around it.

Definition 4.2 (Message)**.** The data that an IoT device transmits to other devices.

Definition 4.3 (Message rank)**.** An identifier value that is associated with each message that corresponds to the criticality of data contained within the message itself.

Definition 4.4 (Device's behavior)**.** A value associated with each device that classifies how a device behaves for a given message rank.

Definition 4.5 (Behavioral vector)**.** A collection of behavioral values that uniquely represent the behavior of a device for all available message ranks.

Definition 4.6 (Device's trust level)**.** A calculation of the behavioral vector that is represented as a single value for the overall trust of a given device.

Our problem statement can be summarized as follows: Given an IoT-based system setup, our goal is to design and implement a dynamic trust management service through evaluating the trustworthiness of the IoT devices considering the messages being transmitted and their ranks along with the devices' behavior in the system. This service will facilitate the ability to reduce the impact of malicious actors manipulating the system and boost the confidence level in the data being collected.

5. Threat Model

Malicious attacks can impact how IoT entities communicate with each other and can affect trust building in a given network, which could have a negative impact on the system. For instance, in IoT-based connected vehicles setup, these issues could lead to negative effects on human safety.

In this study, we focus on a set of threats with different targeted effects on a system:

Threat 1 – Breakout Fraud: A breakout fraud attack occurs when a device attempts to first build trust in the network by providing accurate information, but at a given point in time, it begins acting maliciously by transmitting false data to other devices. This attack aims to take advantage of the fact that the device has earned a high trust value, implying that other devices in the network will most likely accept its data. At such a juncture, the malicious device will be able to successfully inject malicious data into the system, which can negatively impact the system's operations.

Threat 2 – Selective Behavior IoT devices can act maliciously with a high probability that they randomly switch from sending accurate messages to inaccurate messages and back again. Selective behavior attacks seek to maintain a high trust value such that, upon injecting malicious data, it is more likely for the data to be accepted (other devices will apply this data to a decision-making process). However, upon evaluating these messages, the device's trust will be lowered. Thus, the device will opt to switch back to sending accurate messages for a short period of time.

Threat 3 – Illusion-Based Attacks Illusion-based attacks represent a threat that occurs when devices have knowledge of the data that relates to critical messages. With that knowledge, malicious devices can create the illusion of a critical incident by forcing their data to model those critical messages. This attack is especially dangerous since it has the potential to deceive protocols that are in place to prevent the injection of malicious data. For example, by creating an illusion-based attack and circumventing protection protocols, the smart-city system would accept the data and be inclined to make decisions based on this new information. But, because this information is actually inaccurate, the smart city or connected vehicle network could potentially make decisions that would negatively impact the overall condition of the network.

Threat 4 – Colluding Attacks In traditional trust management implementations that use a majority-based consensus mechanism [23], there is the potential for a colluding attack or an overrule of the majority attack. This occurs when a number

Table 2: Communication messages are categorized into ranks based upon the severity of the information present in a given message. Four ranks are presented, starting from rank 0 representing no information to be shared, all the way to rank 3 providing critical information that alerts others to the detection of a car crash.

Rank	V2V Message
0	**General alert –** Corresponding to specific sensor data such as rain detection
1	**Object on road –** Incident message informing others of an object detected on the road
2	**Emergency vehicle –** Message regarding the detection of an approaching emergency vehicle
3	**Car collision –** Incident message alerting others of a car collision

of malicious devices reach consensus and the totality of those devices' trust values outweighs the actual trustworthy devices (such that the malicious information is accepted as accurate). One example is as follows: one device has a 100% trust value while four other malicious devices each has 26% trust. If the four devices with lower trust agree, then their weighted trust values are higher than the trust value of the one telling the truth; thus, the consensus has now been compromised. This is a serious problem with traditional trust management approaches. Specifically, smart cities and connected vehicles rely on the system being able to make real-time decisions when necessary. By not being able to detect the presence of malicious devices in the network, malicious devices in turn will artificially inflate their trust values, which can be used to propagate malicious data throughout the network.

5.1. Impact Level

To supplement the threat descriptions, we summarize the threat impact surfaces in Table 1.

- **Misleading Impact:** This impact is classified as having an overall misleading effect on the system. This could be because of an artificially inflated trust value or a message that deceives other devices in the system into reacting in a certain manner.
- **Response Accuracy Impact:** These threats seek to communicate specific information such that when processed, the system will respond inaccurately to the events that are truly occurring.
- **Immediate System Impact:** Threats that have the potential to quickly and severely harm the system are said to have an immediate system impact. Usually, this will occur when the malicious device intends to cause as much damage as possible while knowing that it will be detected and removed from the system quickly.
- **Prolonged System Impact:** Threats that remain in the system won't necessarily cause immediate harm but instead seek to remain in the system until such a point that it will have the potential to cause a significant impact over a longer period of time.

Algorithm 1: Behavior-based Enhanced Trust management System

V ← set of all vehicles
Rank ← set of all ranks
$MR \leftarrow [\,]$
for $v \notin V$ **do**
    Initialize(v)
for $v \in V$ **do**
    $(m_v, r_{m_v}) \leftarrow$ generateMessage(v)
    $MR \leftarrow (m_v, r_{m_v})$
for *each* $(m_i, r_{m_i}) \in MR$ **do**
    $O_{elo} \leftarrow$ calculateOpponentElo(V, r_{m_i})
    $nn_r \leftarrow \mathrm{NN}(m_i)$
    if $r_{m_i} = nn_r$ **then**
        $\vec{B_v}(i) \leftarrow$ Update(True, O_{elo})
    else
        $\vec{B_v}(i) \leftarrow$ Update(False, O_{elo})
$MR \leftarrow \emptyset$

6. Behavior as a Service for Trust management in IoT Devices (BEAST)

Our approach is carried through four consecutive phases[2] as depicted in Fig. 1. Starting with the ***Data Collection and Behavioral Model Construction*** phase, which presents how devices collect their data and how the data is compiled to form the behavioral model. The second phase is the ***Behavioral Pattern Identification Process*** that presents how deep learning is used to classify devices' behavior. Following that is the third phase, ***Behavioral Assessment***, which describes how the accuracy of the message is used to develop a novel Elo-based behavioral vector that will be used in the following phase [58]. The last phase is ***Trust Modification***, in which each device's new trust value will be derived from the behavioral vector. Our approach is summarized in Algorithm 1.

In this work, we chose connected vehicles as our case study to demonstrate the applicability of our proposed framework.

Phase 1 – Data Collection and Behavioral Model Construction:

6.1. Data Collection

This phase focuses on collecting driving statistics data of each vehicle within a localized geographical region. This includes each vehicle's driving speed, acceleration, braking rate, etc (See Table. 3). Additionally, data to be collected includes surrounding information reported by vehicles such as road conditions, which will be transmitted as ranked messages, which are presented

[2] An assumption is made that devices participating in this system have undergone an enrollment procedure [56, 57] to verify the identity of the device as well as provide the necessary application components (described in this section) such that the device is able to contribute.

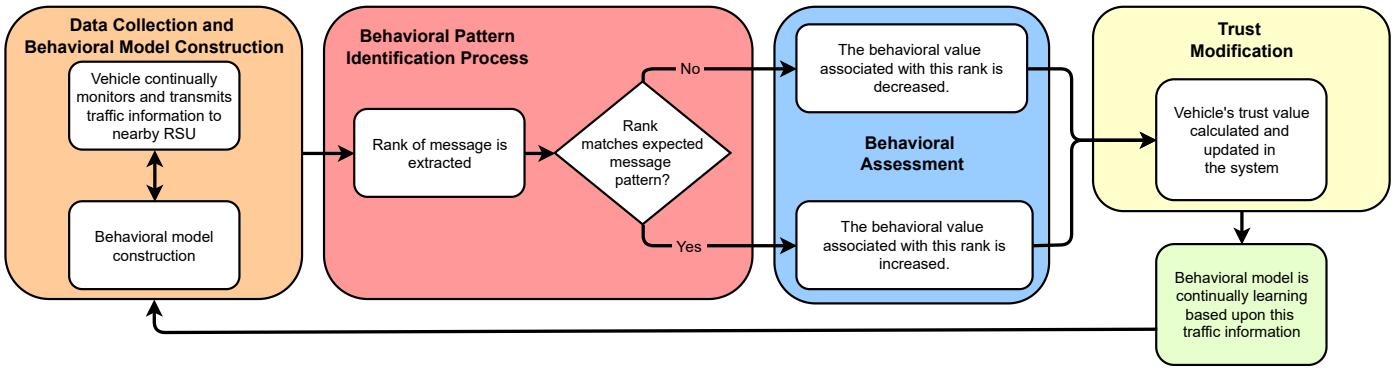


Figure 1: Multiple consecutive phases of the Behavior-based Enhanced Trust management System (BEAST)

in Table 2. These ranks will be used to reflect how critical the situation being reported is, where higher-ranked messages will represent a more critical situation compared to that with lower-ranked message. The focus is to collect data that is informative and leads to a safer environment.

The driving statistics and associated ranks will then be propagated to a nearby RSU for further processing. This is done to alleviate the load on the connected vehicle from processing data while allowing the RSU to conduct a deeper analysis of the collected data. As data is supplied, the RSU will collect and compile the driving statistics and apply deep learning algorithms to construct a neural network that will represent a behavioral model that is the foundation for the next phase. This behavioral model will accurately represent how vehicles in the localized geographical region typically behave in regard to the driving statistics under specific conditions.

6.2. Behavioral Model Construction

In this study we utilized the TensorFlow library to build a sequential feed-forward neural network that acted as the behavioral model. Table 3 represents a few data points that were used, from which we extracted seven inputs (excluding the timestamp). Additionally, four outputs were identified and are described in Table 2.

In this work, we follow traditional design by implementing the rectified linear activation function ReLU for the hidden nodes as given in Eq. 1, where x is the input to a neuron [59]:

$$y = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases} \tag{1}$$

The softmax activation function was used only by the output layer as a way to perform multi-class classification [60]. The equation to calculate the sigma of the softmax activation function is presented in Eq. 2, were z_i is the input vector, e^{z_i} is the standard exponential function for the input vector, K is the number of classes in the multi-class classifier, and e^{z_j} is the standard exponential function for the output vector:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K \tag{2}$$

Table 3: SUMO simulation data snapshot showing the simulation time (sec), the vehicle ID, the vehicle's position, speed, location (x,y), acceleration, speed, brake rate and the message being reported by that vehicle

timestep_time	0	1	1
vehicle_id	0	0	1
vehicle_position	5	7	5
vehicle_speed	0	2.12	0
vehicle_x	2841.33	2840.47	604.42
vehicle_y	2724.56	2722.62	2961.69
motionState_acceleration	0	2120	0
motionState_speed	0	212	0

Another critical component we considered when designing a neural network is the loss function, which was used during training such that it can be minimized, generally to yield higher accuracy. For the purposes of this work, Tensorflow Kera's categorical cross entropy was selected as it is a common used loss function for multi-class output data [61]. The equation for categorical cross entropy is provided in Eq 3, where M is the total number of classes, y is the indicator if class c is correct for observation o, and p is the probably that o is of type c:

$$-\sum_{c=1}^{M} y_{o,c} \ln(p_{o,c}) \tag{3}$$

The last component needed is the optimizer. Optimizers essentially determine how biases, weights, and loss functions work together to increase the overall accuracy of the model, and Adam is generally considered the best algorithm and was used in this design [62]. Adam essentially focuses on modifying the gradient decent algorithm to use the average of the gradients. This allows the model's loss to be minimized quicker than traditional methods [63].

Knowing our inputs and outputs, activation functions, loss algorithms, and optimizations it is possible to brute-force the architecture of a neural network to yield sufficient accuracy to satisfy the requirements of this study.

This neural network was trained via an iterative approach (presented in algorithm 2) where a trivial model was constructed with only the input layer, one hidden layer with one neuron, and

the output layer. The model was compiled, fit, and an evaluation was performed with regard to loss and accuracy. The second step was to increment the number of neurons in the hidden layer. This brute-force methodology was followed until there was a model with five hidden layers and 100 neurons per hidden layer. The model that yielded the best loss and accuracy was saved and used through the remainder of this study. The algorithm described above is presented in Algorithm 2.

Algorithm 2: Iterative Neural Network Development

```
Let NN be the neural network model
input_units ← 7
output_units ← 4
for hidden layer i < 5 do
    for number neurons n < 100 do
        Create NN with n neurons to i hidden layers
        fit(NN)
        evaluate(NN)
```

Phase 2 – Behavioral Pattern Identification Process: The behavioral model represents how vehicles typically behave in a given area. In this phase, the incoming vehicle data will be input into the behavioral model to calculate the expected rank of the message. This will allow for a comparison of the vehicle's driving statistics to the expected behavior.

Malicious vehicles will often inject data into the network with the purpose of causing havoc; malfunctioning vehicles will not realize they are submitting inaccurate information, which makes the detection of these intended or unintended threats in real-time critical. These devices must be punished and effectively removed prior to causing any harm. By monitoring the exchange of data and using deep learning-based behavioral model, the vehicle's data can be classified to determine whether the driving statistics of the device matches the rank based on the behavioral model. If the driving statistics do not match, then the model will output a classification for the message that implies the vehicle is malfunctioning, being malicious, or for any of the following reasons:

- A malfunctioning sensor would produce incorrect readings. For instance, connected vehicles would send data that does not match their driving pattern [64].
- Injection of falsified information can negatively impact the system's abilities to make the best decisions. A malicious vehicle could be attempting to inject data to have subsequent vehicles rerouted so it would have the road to itself [65].
- General malicious behavior can alter the system's ability to detect truly malicious behavior. For example, driving behavior that does not comply with the road rules (such as speed limit) could impact the ability to determine the standard behavior of the road [66].

In the event that the behavioral pattern identification process relates the driving statistics back to the original rank, the vehicle reported, then the message is deemed accurate and trustworthy, thus increasing the vehicle's behavioral value for that rank. But, if the message rank is different than what the vehicle reported, the message is deemed to be an anomaly and thus the vehicle's trust value will be decreased accordingly. Additionally, messages that are found to be trustworthy will be appended to the behavioral model training set, such that over time behavior has the opportunity to change and will not become stale.

Phase 3 – Behavioral Assessment:

There are two general means of calculating Elo, or the rating of a player or in our implementation a device's trustworthiness [58, 67]. The original Elo system was developed by Arpad Elo in the 1950s to calculate the strength of chess players [58]. This model applies a statistical methodology that takes into account the opponent's strength to determine the strength of the player, this methodology will be discussed in more detail below. The second means of calculating Elo is called the Glicko system, developed by Mark Glickman [67]. This system was designed as a way to more accurately reflect new players' ratings, the idea being that newer players are more likely to increase their skill level more quickly than players who have played thousands of games. Thus the Glicko system puts extra emphasis on the number of games played.

Algorithm 3: Initialize(v)

```
Rank ← set of all ranks
B⃗_v ← |B_0, B_1, B_2, ..., B_{Rank−1}|
for r ∈ Rank do
    B_r ← 800
V ← v
```

For the purposes of this work, it was determined that the original implementation of Elo is best suited due to the fact that allowing new devices to more quickly build trust means that the system is more vulnerable to breakout fraud. And in this regard, the opponent's Elo was calculated as shown in Algorithm 4. As devices enter the system, they will be considered neutral entities that will each have a neutral trust value, meaning an initial value that all devices begin with that is not too high or low such that the device has the ability to both build and lose trust. Further, upon introduction to the system, each vehicle will have a behavioral vector initialized (Algorithm.3). This behavioral vector will consist of n unique behavioral values that correspond to how a specific vehicle's behavior corresponds to a certain rank, where n is the number of possible message ranks as presented in Table 2. The implementation of this behavioral value will be via an application of the Elo rating system [58].

Algorithm 4: calculateOpponentElo(V, m_r)

```
T_R ← 0
for each v ∈ V do
    T_R += B⃗_v(m_r)
O_elo ← T_R/|V|
return O_elo
```

As messages shared by a vehicle are analyzed and evaluated through the behavioral pattern identification process, the behavioral vector's indices (b_i) represent that device's trustworthiness level for the given rank or index and will be continuously increased or decreased according to how the vehicle's behavior corresponds when that rank is applied to a message.
The behavioral vector ($\vec{V_b}$) consists of n behavioral values for a given rank (b_i) such that

$$\vec{V_b} =< b_0, b_1, b_2, \ldots, b_n > . \tag{4}$$

Each index of behavioral value is calculated via the same mathematical formulas of the Elo system after being initialized, as shown in Algorithm 3 [58].

Algorithm 5: Update(Result, R_b)

$R_a \leftarrow$ behavior of device a
$R_b \leftarrow$ be the behavior of device b
$E_a \leftarrow$ expected outcome, that is defined as the following:
$E_a \leftarrow 1/(1 + 10^{(R_b - R_a)/400})$
if *Result* **then**
    $S_a \leftarrow 1$
else
    $S_a \leftarrow 0$
$K \leftarrow 32$
Let R'_a be the new behavior of device a, be defined by:
$R'_a \leftarrow R_a + K(S_a - E_a)$
Return R'_a

Eq. 5 represents the likelihood that *device A* will tell the truth. This equation takes into account the rating of *device A* (R_a) and the rating of *device B* (R_b) to make this calculation:

$$E_a = 1/(1 + 10^{(R_b - R_a)/400}). \tag{5}$$

As we consider the rankings of *device A* and *device B*, we will be able to calculate the expected Elo score for *device A*, which reflects the likelihood that *device A* is telling the truth.
Eq. 6 shows the formula used to update *device A's* rating. Where R'_a is the new rating of *device A*. R_a is their current rating. K is the multiplier for a change in Elo, and S_a is the actual score *device A* achieved (1 if they win or the message was benign, 0 if they lose or the message was malicious). This is to say that if a device were to have the behavioral pattern identification process to determine if the message was true and accurate, then the behavioral vector with the index equal to the rank associated with that message will be increased by Eq. 6. Conversely, if the behavioral pattern identification process determines the message to be inaccurate or malicious, both the predicted rank of the process and the rank of the message the vehicle sent will be decreased in accordance with the following equation. The decrease for both the predicted rank and the message rank aligns with how difficult it is to build and lose trust, as presented in [68]. This process is presented in Algorithm 5.

$$R'_a = R_a + K(S_a - E_a) \tag{6}$$

Phase 4 – Trust Modification: As previously mentioned, the behavioral vector will be the instantaneous representation of the device's behavior for all available ranks; however, this behavioral vector must be integrated with trust to achieve a more accurate representation of the device's trustworthiness. After determining whether the device has matched the expected behavior or is an anomaly (either malfunctioning or malicious), the RSU will calculate the device's trust value with respect to the 2-norm, as this is a standard method of measuring a vector as given in equation 7:

$$T = \|\vec{V_b}\|_2. \tag{7}$$

6.3. Generality discussion

The steps described above can be applied to any system that is driven by peer communication and relies on the evaluation of this data. Additionally, any of the steps can be modified in place as long as the primary goal of each phase is accomplish. That is, BEAST is designed as a service acting as a framework where the specifics of each phase are suggestions but can be interchanged if desired by the system designer. Of the five components of this design, phase one (data collection and behavioral model construction) is the most fundamental upon which the remainder of the system is built. When applying this design to a new system, the only requirement is to have taken the necessary steps to ensure data collection is in place and construct a behavioral model that is able to accurately reflect the behavior of devices. Once this has been conducted, the final setup requires routing the data being transmitted through the behavioral model for evaluation. The output of this behavioral pattern identification process (evaluation of a message as benign or malicious) is used in the latter phases (three through five) to increase or decrease the trust of a device and label it as trustworthy or untrustworthy.

Additionally, to promote the longevity of this design, it is critical to enable the behavioral model to be updated to support the growth and changes in the network. This step is represented in our feedback loop in Fig. 1. Without this feedback loop, the behavioral model would become stale and the nature of the system would change the behavioral model would flag every interaction as an anomaly and thus all devices would be considered untrustworthy. To further support this design as a service, the method and frequency of updates to the behavioral model as well as recompiling and distribution to devices is configurable by the application designer.

7. System Design and Evaluation

As previously mentioned, this approach can be applied to any system that focuses on peer communication and data evaluation. Thus, for the purposes of this study, the team selected a use case of connected vehicles. This was selected since BEAST can reliably evaluate the data that vehicle to vehicle (V2V) communication shares, thus greatly increasing the overall safety of smart cities where these connected vehicles are deployed. This is supported by the National Highway Traffic Safety Administration (NHTSA); they describe V2V communication messages as basic

Table 4: Relation of rank to the expected driving pattern described below. These vehicle speeds and braking rates are SUMO-specific values that are uniquely represented by the simulation. The values used to identify each rank, is derived from the distribution of data points from over 500 unique simulations.

Rank	Driving Patterns
0	**Vehicle in motion:** Vehicle is traveling without impact
1	**Braking occurs (1-2):** Vehicle speed must be less than 1200
2	**Heavy braking (2-3):** Vehicle speed must be less than 800
3	**Major braking (3-4, 5):** Vehicle speed must be less than 400

safety messages (BSMs) that correspond to messages passed between vehicles regarding dynamic information such as heading, speed, and location as well as warning messages that alert other vehicles of dangerous circumstances [8]. Additionally, there are other safety applications that depend on the sensors with which connected vehicles are equipped [69]. Potential messages or warnings include the following (as defined by the NHTSA [8]):

- **Intersection Movement Assist (IMA):** Alerts drivers when it is unsafe to merge into an intersection.
- **Left Turn Assist (LTA):** Warns the driver that it is unsafe to make a left turn because there is oncoming traffic and thus a potential for collision.
- **Emergency Electronic Brake Light:** An alert that triggers when a driver is applying the brakes. This is useful when a lead vehicle might not be visible to a trailing vehicle because of a blind curve or severe weather conditions.
- **Forward Collision Warning:** Warns the driver of potential collision with a leading vehicle. Such a warning would also be beneficial to other drivers and infrastructure because it is a measurement of how close vehicles are following.
- **Do-Not-Pass Warning:** Communication warning a following vehicle that it is not safe to pass. This is attributed to a number of reasons (e.g., oncoming traffic is approaching, so it is not safe to pass at present).

Because connected vehicles is a relatively new research area, and few exist in production environments, real-world data is not readily available for the demands of this work. Simulation of Urban Mobility (SUMO) and OpenStreetMap (OSM) were used to design and simulate connected-vehicle-related scenarios. Large-scale road networks were implemented and designed with an abundance of cars to log and output driving statistics and vehicles' shared messages to form the dataset used for the reaminder of this work [70, 71, 72].

Through the use of SUMO, numerous reported statistic "features" such as speed, position, acceleration, and braking were generated. Samples of these statistics are shown in Table 3.

As mentioned, in addition to the reported features, the collected data will also contain a ranked message that correlates the driving statistics to warning (or BSMs) and incident messages. Correlation of message ranks to a particular communication message is shown in Table 2, and the distinction created by these particular messages is presented in Table 4.

A message rank of 0 does not correspond to any message in particular. This form of message does not have any impact on the system but is implemented such that the vehicle is able to transmit its current driving statistics to the infrastructure so that the smart city is aware of the current statistics of drivers on the road. These types of messages are considered warning messages that represent NHTSA's BSMs. These messages should not warrant any drastic changes in driving patterns. They are primarily for the other drivers and infrastructure to be made aware of the current environment. In real-world applications, these messages could also be derived from another sensor. Specifically, these sensors may include rain detectors, light detection for headlights, detection of ice, or an ultrasonic sensor or camera that is able to detect and identify objects on the shoulder of the road, such as a traffic sign or a pedestrian [73], or even come from a sensor monitoring the acceleration of said vehicle or other information such as the blinker indicating a lane change. There also exist incident messages (ranks 1-3), where it is expected to force a substantial change in the current driving pattern to accommodate the reason for the message. Messages with high ranks indicate the seriousness of the incident, which will impact the system and help efficiently implement traffic rerouting when necessary [74].

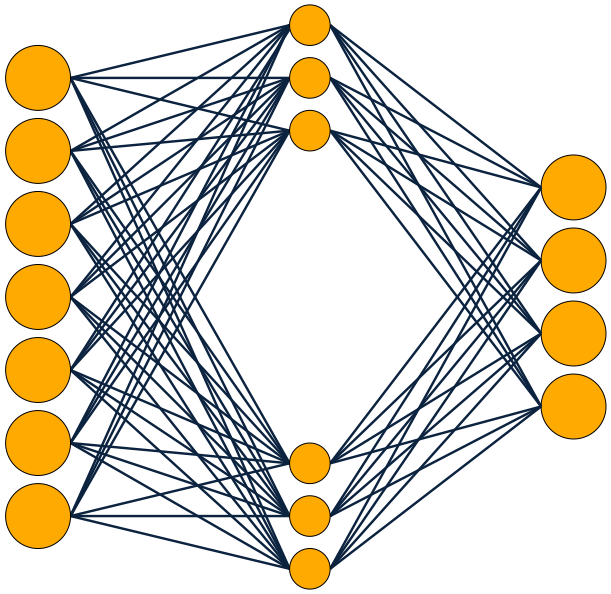


Figure 2: An architectural diagram of the sequential feed-forward neural network used as the behavioral model for this work.

7.1. Behavioral Model Construction Evaluation

The architecture of the sequential feed-forward neural network that is used as the behavioral model for this work is presented in Fig. 2. As previously mentioned, our data set is obtained from several SUMO simulations to develop a baseline of how vehicles would behave in a given geographical area. To describe this behavior, the team captured seven input features: the position of the vehicle within the map or corresponds to a specific road that the vehicle is on, the average speed of the vehicle since the last message, both the latitude and longitudinal coordinates of the vehicle to obtain specifically where the vehicle is on the road, the current acceleration of the vehicle, the current speed of the vehicle, and lastly how hard this vehi-

Table 5: BEAST Neural Network Model Accuracy Evaluation

Rank	Correct Predictions	False Predictions	Accuracy
0	141,662	451	99.68%
1	58,495	211	98.67%
2	15,691	257	98.39%
3	66,202	212	99.68%
Total	**282,050**	**1,131**	**99.60%**

Table 6: BEAST Neural Network Model Time Evaluation

Vehicle type	Number of messages	Time (sec)
Malicious	249	3.53
New vehicle	599	8.66
Threat Model Analysis	249	3.42
Total	1097	15.61

cle is braking at the given moment. This information provides specifics regarding the current position of the vehicle as well as the specific driving statistics. These two components are both necessary for the evaluation of behavior because the location of the vehicle can impact how it behaves (i.e., a vehicle is on the interstate it generally is driving much faster than on a highway) and of course, it is necessary to monitor the driving statistics of the vehicle to insure the vehicle is behaving similarly to other vehicles in the network. Using the iterative approach as described in Section 6.2, the model that yielded the highest accuracy was a model with one hidden layer with 59 neurons (Fig. 2). These neurons used the ReLU activation function and were fed into the output layer with four neurons using the softmax activation function to obtain the class from the result of the neural network.

7.2. Behavioral Model Performance Evaluation

The performance evaluation for BEAST's neural network model is divided into two separate evaluations: an accuracy analysis as well as a time complexity analysis.

The first evaluation metric for BEAST's neural network model is an accuracy analysis. Since this multi-class model is able to predict the rank of the message, it is necessary to compare the actual and predicted values. The specific rank accuracy is presented in Table 5. The model can overall accurately predict message ranks with over 99% accuracy. Table 5 shows how each actual message rank compares to the predicted rank. This matrix shows that rank 0 messages are generally predicted to be rank 0, with less than 0.5% being classified as rank 1. Similarly, rank 1 messages are primarily classified as rank 1, still having only 1.5% being classified incorrectly. This general trend continues, indicating that this model is able to accurately predict message rank values.

The time complexity analysis is shown in Table 6. This table displays how many messages each simulation incorporated, as well as how much time (in seconds) it took to process each simulation. It should be noted that the time presented in this table includes the overhead of vehicles to process the message and transmit it to the central node. Even so, it can be seen that in the malicious and threat model analysis simulation it took approximately 3.5 seconds to process 250 messages, while in the new vehicle analysis simulation it took 8.66 seconds to process almost 600 messages.

Therefore, to process the 1097 messages in 15.61 seconds leads to a processing time of 70.28 messages per second or conversely 0.014 seconds per message[3]. This data shows that this approach is able to meet the necessary real-time and scalability requirements, as a single node is able to process 70 vehicles sending one message every second. To extrapolate this further, if vehicles were to instead send a message every minute instead of every second, a single node could process 4200 vehicles simultaneously without building a queue.

7.3. BEAST Evaluation

To evaluate our proposed system, we designed test use cases to demonstrate the capabilities of our proposed approach to mitigate the threats described in the threat model (Section 5).

Our evaluation is demonstrated via three use cases representing different scenarios in the system. The base use case is where vehicles are participating in the system and being evaluated as behaving normally (benignly) or maliciously. The new vehicle use case demonstrates how our proposed approach addresses new devices entering the system and how trust is gained, and the threat model use case covers each of the threats presented in Section 5, as well as how our proposed approach handles that threat.

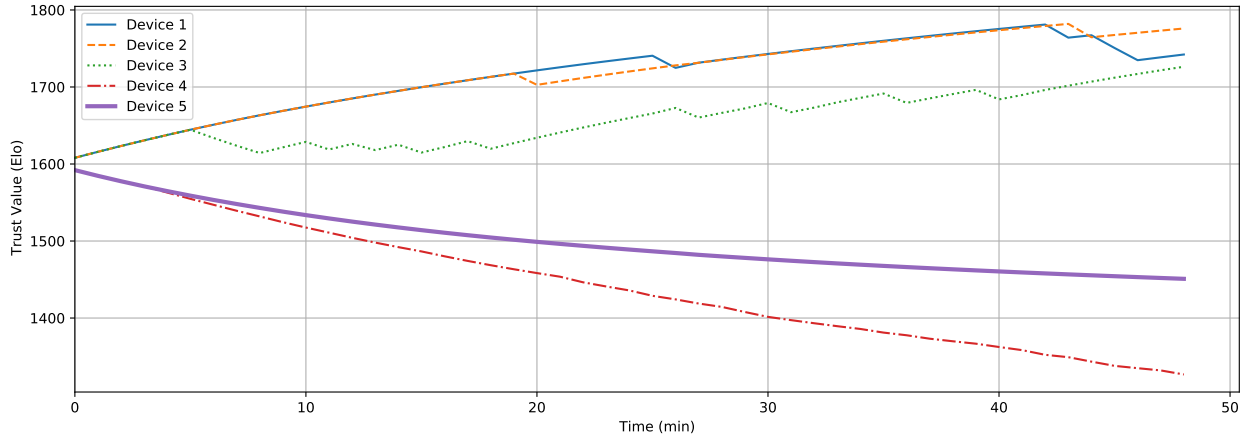
Each of these use cases was simulated to contain five vehicles, where each vehicle remained in the system for at least 50-time units (minutes) such that enough time has passed for the messages to be transmitted, processed, and trust values to be modified thoroughly enough to gain an accurate simulation of the events that occurred.

7.3.1. Base use case

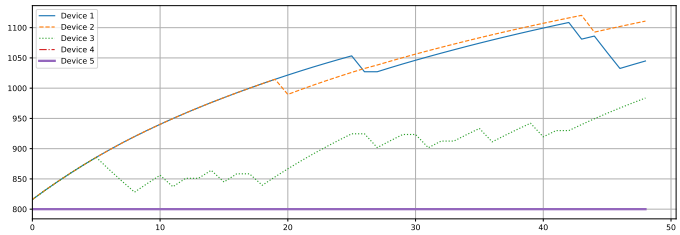
This use case demonstrates how both trustworthy and malicious devices will transmit messages and have their behavioral values modified across their lifetimes. This is clearly shown in Fig. 3a as devices 1, 2, and 3 have their trust values trending upward with a few minor decreases, which can be attributed to an event such as the vehicles driving too fast for the indicated message rank.

Looking at vehicles 1, 2, and 3 specifically, it can be seen in Fig. 3b and Fig. 3c that their behavior for these two ranks steadily increases, indicating that the devices were transmitting lots of rank 0 and rank 1 messages. Further, these messages were deemed accurate by the deep learning neural network model, increasing their behavior for these ranks accordingly. Conversely, looking at Fig. 3d and Fig. 3e, it can be seen that the behavior for rank 3 did not increase but instead maintained their current behavioral values. This is because the vehicles did not send messages with this rank and thus did not have their behavior adjusted. These trust values presented in Fig. 3a are calculated via the equation presented in Eq. 7, or the 2-norm of the behavioral values for each message rank. This shows that even if vehicles

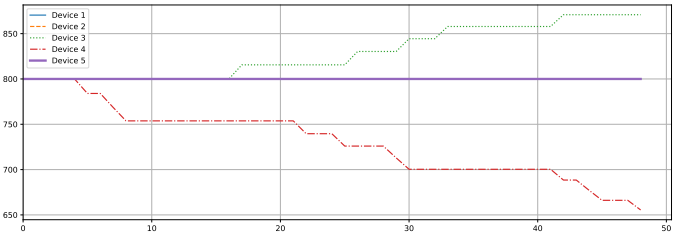
[3]This performance resulted from an AMD Ryzen 5600X x86-64 processor at 3.8GHz and NVIDIA GeForce 2080Ti.



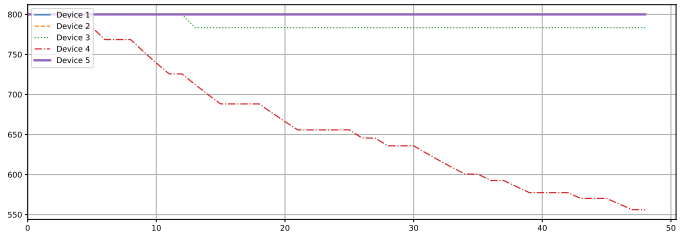
(a) Overall trust value changes over time driven by the behavioral values. The x-axis is represented by time in minutes, which is equivalent to the number of messages sent, while the y-axis corresponds to the device's trust value represented by Elo.



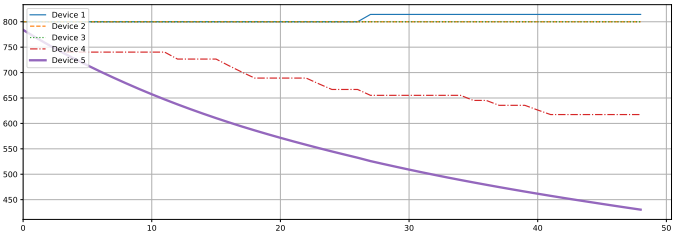
(b) Behavior of devices for rank-0 messages



(c) Behavior of devices for rank-1 messages



(d) Behavior of devices for rank-2 messages



(e) Behavior of devices for rank-3 messages

Figure 3: Base use case: results from a simulation with five devices, three of which have benign behavior and two of which are behaving maliciously. The x-axis is the time of the simulation (minutes), while y-axis represents the trust value of the device (Elo).

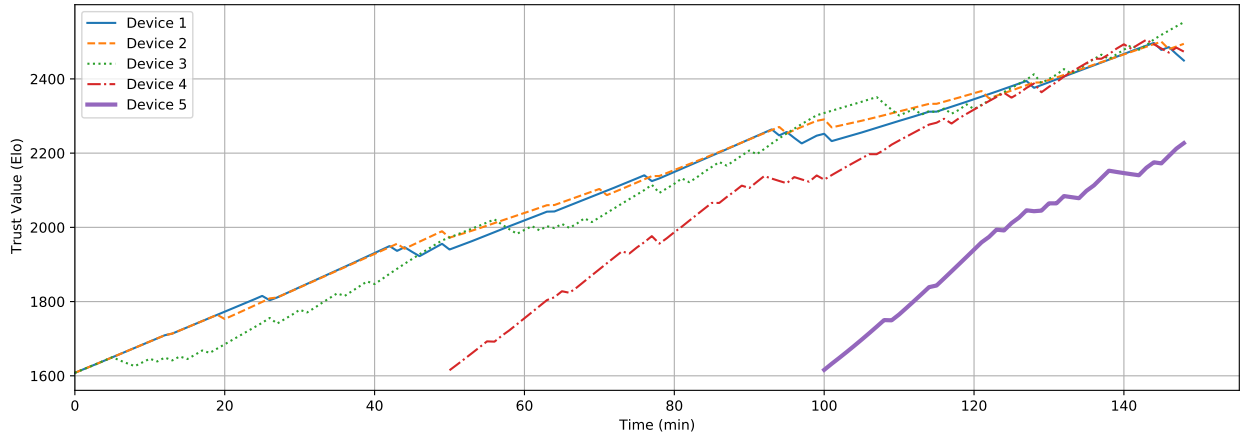
are to only send messages of a specific rank, it will impact their overall trust value while not impacting the behavior value for ranks, for which the vehicle did not send messages.

Looking at the two malicious devices (vehicle 4 and vehicle 5) it can be seen in Fig. 3a that their trust values steadily decrease throughout the simulation. Specifically looking at Fig. 3b, we can see that vehicle 4 and 5 had no changes, indicating that they did not send any messages with rank 0. Similarly, in Fig. 3c we can see that vehicle 4 had several decreases indicating that all of the messages the vehicle sent were deemed to be malicious and thus have its behavior decreased. With ranks 0, 1, and 2, vehicle 5 had no changes to its behavior, indicating it sent no messages, while vehicle 4 had several messages sent. However, vehicle 5 sent several messages of rank 3 all of which were determined to be malicious and thus negatively impacted vehicle 5's behavioral value and overall trust value. Looking at the behavioral values represented by Figs. 3b, 3c, 3d, and 3e, it can be seen that they directly correspond to the vehicle's trust values presented in Fig. 3a.

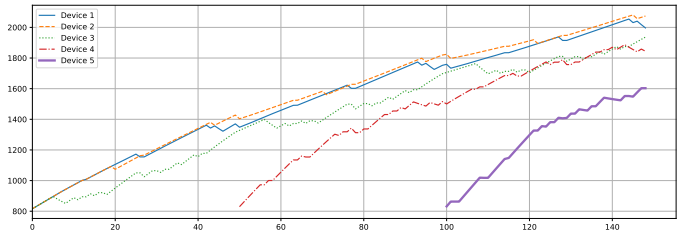
7.3.2. New vehicle use-case

Our results for this case are shown in Fig. 4. These results demonstrate the rate at which the vehicle's trust values can increase across the lifetime in the system. This simulation contains five total vehicles. Three (vehicles 1, 2, and 3) entered the simulation at time zero with a trust value of approximately 1600 (the 2-norm of a vector length 4 with values of 800). It can be seen in the behavioral value figures (Figs. 4b and 4c) that these vehicles primarily send rank 0 and rank 1 messages as their behavioral values drastically increase over this time frame, while rank 2 and rank 3 (Figs. 4d and 4e) primarily maintain their initial behavior values, or had slight decreases which can be explained by the vehicle potentially driving too fast for the given message. Again, according to the method for calculating the trust value, it is just the 2-norm of the behavioral value vector (represented by Eq. 7). It is clear that the these vehicles' trust values also increase.

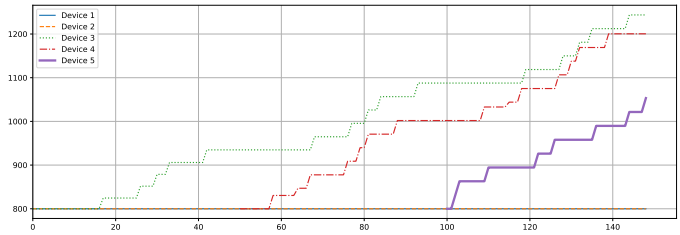
It can be seen in Fig. 4a that at time 50, a new vehicle (vehicle 4) enters the system with the original neutral trust value of approximately 1,600, at which point it will closely mimic



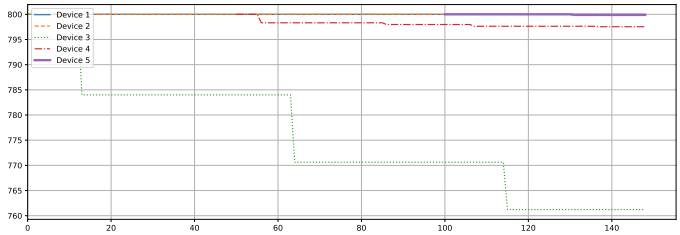
(a) Results from a simulation with five devices, three of which have a benign behavior and two of which are behaving maliciously. The x-axis is represented by time in minutes, which is equivalent to the number of messages sent, while the y-axis corresponds to the device's trust value represented by Elo.



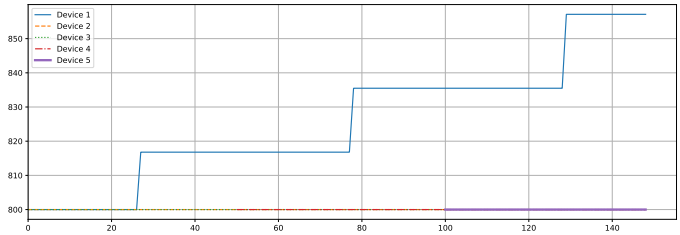
(b) Behavior of devices for rank-0 messages



(c) Behavior of devices for rank-1 messages



(d) Behavior of devices for rank-2 messages



(e) Behavior of devices for rank-3 messages

Figure 4: New Vehicle: The x-axis is the time of the simulation (minutes), while they-axis represents the trust value of the device (Elo).

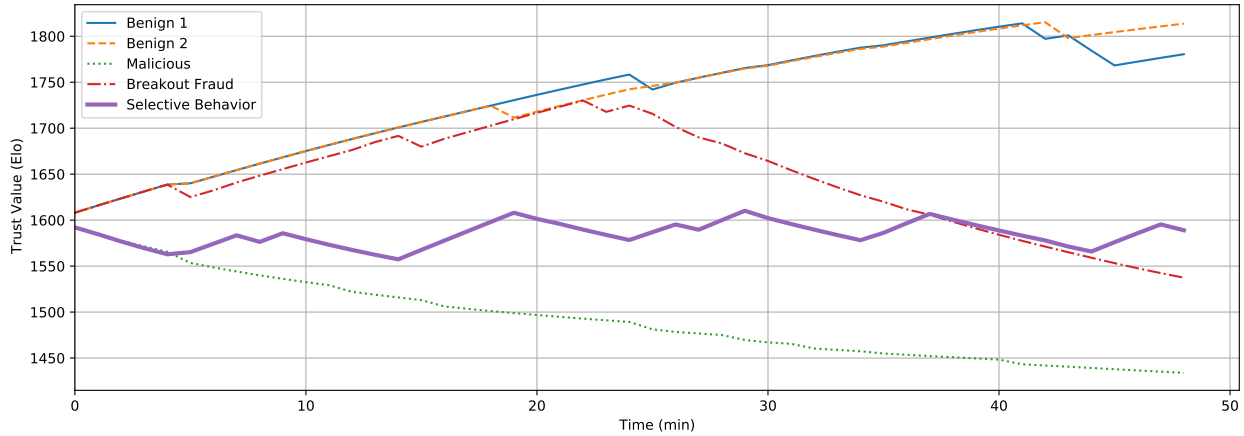
what was described above where it will send accurate messages of rank 0 and 1, allowing its overall trust value also to increase. This process is repeated again at time 100 with another vehicle (vehicle 5). Overall, it can be seen that all trust values are generally trending upward, indicating benign behavior, but the critical part to note is the rate at which each of the vehicles' trust is increasing. When comparing the rate of trust value increase of vehicles 1, 2, and 3 to that of vehicle 5, it becomes clear that vehicle 5 is increasing significantly faster. This is because the method for increasing or decreasing a vehicle's behavioral value is using the Elo rating system. As was discussed in Section 6 - Phase 3, the 'opponent' vehicles (vehicle 1, 2, 3, and 4) already had higher trust values, so vehicle 5 was able to effectively play against higher rated devices and thus increase its Elo more quickly than vehicle 1, 2, and 3 were able. Because the 'opponents' values, in that case, were equal to their own, the Elo increase was not as drastic.

The overarching idea is that even when some vehicles have existed in the system longer, new vehicles entering the system will be able to gain trust quickly enough that they are able to aid the system.

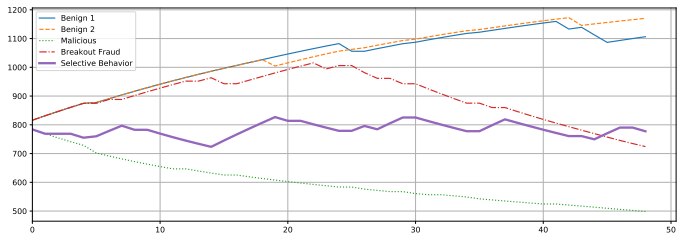
7.3.3. System under the threat model use case

This simulation introduces the threats that were presented in Section 5 with two trustworthy devices, one behaving maliciously, one performing breakout fraud, and another performing the selective behavior attack. Our results for this case are shown in Fig. 5. Fig. 5a presents the overall trust values (Elo) of devices throughout this simulation, and it can be seen that as in all other use cases, vehicles 1 and 2, which are behaving in a trustworthy manner, are able to successfully increase their trust values primarily through sending accurate rank-0 and rank-1 messages, as can be seen in Figs. 5b and 5c.

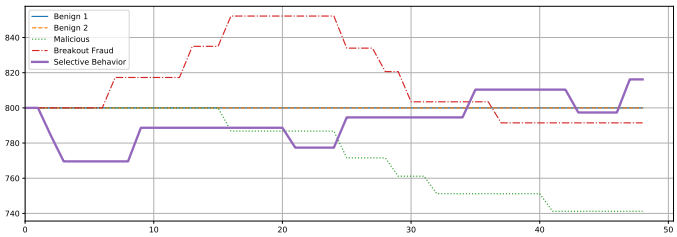
Specifically, looking at the malicious vehicle, it can be seen in Figs. 5a that the trust value is steadily decreasing. The simulation was designed in such a way that the anomalies were injected



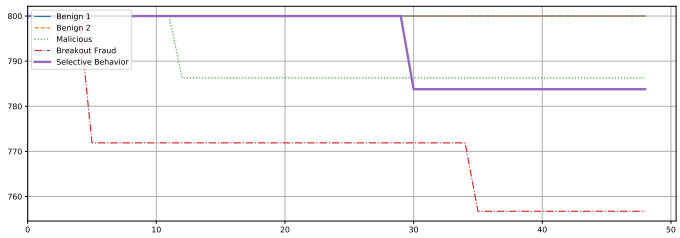
(a) A simulation containing the primary threats from our threat model to analyze how BEAST mitigates these threats. The x-axis is the time of the simulation (minutes), while the y-axis represents the trust value of the device (Elo).



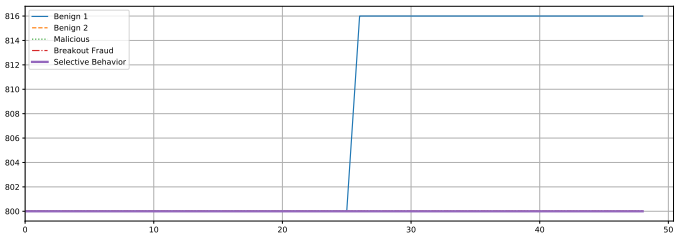
(b) Behavior of devices for rank-0 messages



(c) Behavior of devices for rank-1 messages



(d) Behavior of devices for rank-2 messages



(e) Behavior of devices for rank-3 messages

Figure 5: Results from a simulation with five devices, two of which have a benign behavior and three of which are behaving maliciously, each performing their own attack—specifically, being malicious, committing breakout fraud, and performing a selective behavior attack. The x-axis is the time of the simulation (minutes), while the y-axis represents the trust or behavioral value of the device (Elo).

into the simulation such that the vehicle only sent messages of rank 3; however, according to the behavioral values in Figs. 5b and 5c, it can be seen that the neural network determined these messages to actually be of ranks 0 and 1, thus decreasing those behavioral values accordingly instead of decreasing the rank that the vehicle sent.

Similarly, looking at the vehicle performing breakout fraud (where the vehicle attempts to quickly build trust and then switch to being malicious such that its messages will be trusted, as mentioned in Section 5), we can see that the vehicle gains trust similarly to both of the benign vehicles, but at the halfway point of the simulation the vehicle suddenly begins acting maliciously and sends only malicious messages (again of rank 3). It can be seen that, initially, the breakout fraud vehicle sends several rank-0 and rank-1 messages (Figs. 5b and 5c). Later, when the behavior changes from good to bad, the vehicle starts to send higher ranked messages, but again the neural network determines these messages to have an actual rank of 0 or 1, thus decreasing that behavioral value as can be seen in Figs. 5b and 5c).

Last is the vehicle performing the on-off attack (where the vehicle will periodically switch behavior from benign to malicious, as mentioned in Section 5). Fig. 5a shows that the vehicle has its trust value increasing and decreasing periodically. Looking at the specific behavior values, we can see that the vehicle primarily sent rank 1 and rank 2 messages, since that is the primarily behavior value with fluctuation.

These results show that the methodology introduced in this paper is able to successfully mitigate these attacks by ensuring devices which perform them are punished because the vehicles that are performing the attacks discussed in Section 5 are not able to increase their trust in any meaningful manner.

8. Conclusion

In current trust management approaches, when a majority of devices in the network are malicious and injecting inaccurate data into the system, there is no method that validates the data that these devices are sending. Instead, the system solely relies on the consensus of data to determine accuracy and update the trust value accordingly. This design sought to mitigate the weaknesses from which current trust management systems suffer. These weaknesses include the punishment of malicious actors in real-time as well as mitigating colluding attacks, which seek to exploit traditional consensus mechanisms.

This work has shown that the collection of a device's data enables the creation of a behavioral model that describes how devices in a given geographical location behave; using a deep learning model, it is possible to detect malicious behavior in real time and with over 99% accuracy; and, through this detection process, a behavioral vector can be obtained implementing an Elo rating system to represent the device's trustworthiness for a given message rank. The 2-norm of this behavioral vector was calculated and equates to the device's trust value.

Further, the behavior-as-a-service trust management approach was designed as a framework, which provides three distinct services to trust management: the construction of a behavioral model, a behavioral pattern identification process, and an Elo-based behavioral vector for trust management. Through detailed data collection, a behavioral model can be designed for a localized area, which accurately represents the standard behavior of devices within the area. Then a behavioral pattern identification system was constructed, which utilized a deep learning model such that when new messages from devices were input, the data would be extracted and compared to the behavioral model that was previously created. Through this analysis of data, any devices whose data did not match the standard or expected behavior will be classified as an anomaly and will be punished accordingly. Lastly, after the behavioral pattern identification system, a behavioral vector would be calculated. This behavioral vector is comprised of behavioral values each representing a device's trustworthiness for a given message rank. These behavioral values are based upon an Elo rating system, a proven method for accurately rating skill or, in this case, trust.

The results in Section 7.3 demonstrate that, using the methodology described above, we were able to successfully design a mechanism for monitoring and calculating a devices trust value. Additionally, we were also able to mitigate threats including: breakout fraud, selective behavior, illusion-based attacks, and colluding attacks in real-time. Fig. 5 clearly demonstrates that devices whom are behaving maliciously are punished accordingly, and devices who commit attacks such as breakout fraud and selective behavior are punished quickly and sufficiently enough such that they are unable to maintain a high trust value that would harm the system.

By designing this approach to be used as a service, the ability to modify parameters used throughout this work is key to the overall success of the implementations. If a designer required a trust management implementation with more steady trust values, then simply changing the K-value in Eq. 6 to be smaller would yield a more consistent trust while simultaneously increasing risk in the system as potential malicious actors would not be punished as quickly. If the application this model were being applied onto required incredibly high data scrutiny, then increasing the threshold for a device to be considered trustworthy would be to say that data will only be accepted from devices who have 95% trust value or higher. By designing a dynamic solution, we facilitate the modification of the values used throughout this work such that the requirements of the application that this solution is applied onto can be met.

Acknowledgment

The authors acknowledge the support of the University of Tennessee at Chattanooga. The research reported in this publication was partly supported by the Center of Excellence for Applied Computational Science and Engineering competition.

This work was also supported in part by the National Science Foundation under Grant Nos. 1925603, 1925598, and 1821926. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] N. M. Karie, N. M. Sahri, P. Haskell-Dowland, Iot threat detection advances, challenges and future directions, in: 2020 Workshop on Emerging Technologies for Security in IoT (ETSecIoT), 2020, pp. 22–29. doi:10.1109/ETSecIoT50046.2020.00009.

[2] G. Rajendran, R. S. Ragul Nivash, P. P. Parthy, S. Balamurugan, Modern security threats in the internet of things (iot): Attacks and countermeasures, in: 2019 International Carnahan Conference on Security Technology (ICCST), 2019, pp. 1–6. doi:10.1109/CCST.2019.8888399.

[3] F. Shaikh, E. Bou-Harb, N. Neshenko, A. P. Wright, N. Ghani, Internet of malicious things: Correlating active and passive measurements for inferring and characterizing internet-scale unsolicited iot devices, IEEE Communications Magazine 56 (9) (2018) 170–177.

[4] A. S. Sohal, R. Sandhu, S. K. Sood, V. Chang, A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments, Computers & Security 74 (2018) 340–354.

[5] X. Liu, M. Abdelhakim, P. Krishnamurthy, D. Tipper, Identifying malicious nodes in multihop iot networks using diversity and unsupervised learning, in: 2018 IEEE International Conference on Communications (ICC), IEEE, 2018, pp. 1–6.

[6] S. S. Manvi, S. Tangade, A survey on authentication schemes in vanets for secured communication, Vehicular Communications 9 (2017) 19 – 30. doi:https://doi.org/10.1016/j.vehcom.2017.02.001.
URL http://www.sciencedirect.com/science/article/pii/S2214209616300018

[7] A. Vehicles, Self-driving vehicles enhanced legislation, http://www.ncsl.org/research/transportation/autonomous-vehicles-self-driving-vehicles-enacted-legislation.aspx, accessed on: June 2017.

[8] NHTSA, Vehicle-to-vehicle communication technology.
URL https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/v2v_fact_sheet_101414_v2a.pdf

[9] H. Hasrouny, C. Bassil, A. E. Samhat, A. Laouiti, Group-based authentication in v2v communications, in: 2015 Fifth International Conference on Digital Information and Communication Technology and its Applications (DICTAP), IEEE, 2015, pp. 173–177.

[10] R. Van Der Heijden, Security architectures in v2v and v2i communication, in: Proc. 20th Student Conf. IT, 2010, pp. 1–10.

[11] Z. Yang, K. Yang, L. Lei, K. Zheng, V. C. Leung, Blockchain-based decentralized trust management in vehicular networks, IEEE Internet of Things Journal 6 (2) (2018) 1495–1505.

[12] C. A. Kerrache, C. T. Calafate, J.-C. Cano, N. Lagraa, P. Manzoni, Trust management for vehicular networks: An adversary-oriented overview, IEEE Access 4 (2016) 9293–9307.

[13] M. Blaze, J. Feigenbaum, J. Lacy, Decentralized trust management, in: Proceedings 1996 IEEE Symposium on Security and Privacy, IEEE, 1996, pp. 164–173.

[14] L. Xiong, L. Liu, Building trust in decentralized peer-to-peer electronic communities, in: Fifth International Conference on Electronic Commerce Research (ICECR-5), 2002.

[15] L. Xiong, L. Liu, Peertrust: Supporting reputation-based trust for peer-to-peer electronic communities, IEEE transactions on Knowledge and Data Engineering 16 (7) (2004) 843–857.

[16] F. Kandah, J. Whitehead, P. Ball, Towards Trusted and Energy-efficient Data Collection in Unattended Wireless Sensor Networks, Wireless Networks 26 (7) (2020) 5455–5471.

[17] D. Reising, J. Cancelleri, T. D. Loveless, F. Kandah, A. Skjellum, Radio Identity Verification-based IoT Security Using RF-DNA Fingerprints and SVM, IEEE Internet of Things Journal (2020) 1–1doi:10.1109/JIOT.2020.3045305.

[18] B. Huber, F. Kandah, Behavioral model based trust management design for iot at scale, in: 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), 2020, pp. 9–17. doi:10.1109/iThings-GreenCom-CPSCom-SmartData-Cybermatics50389.2020.00022.

[19] Y. Ren, V. I. Zadorozhny, V. A. Oleshchuk, F. Y. Li, A Novel Approach to Trust Management in Unattended Wireless Sensor Networks, IEEE Transactions on Mobile Computing 13 (7) (2014) 1409–1423. doi:10.1109/TMC.2013.22.

[20] X. Kang, Y. Wu, A Trust-Based Pollution Attack Prevention Scheme in Peer-to-Peer Streaming Networks, Computer Networks 72 (Supplement C) (2014) 62 – 73. doi:https://doi.org/10.1016/j.comnet.2014.07.012.
URL http://www.sciencedirect.com/science/article/pii/S1389128614002667

[21] A. A. Pirzada, C. McDonald, Establishing Trust in Pure Ad-hoc Networks, in: Proceedings of the 27th Australasian Conference on Computer Science - Volume 26, ACSC '04, Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 2004, pp. 47–54.
URL http://dl.acm.org/citation.cfm?id=979922.979929

[22] Z. Liu, A. W. Joy, R. A. Thompson, A Dynamic Trust Model for Mobile Ad hoc Networks, in: Proceedings. 10th IEEE International Workshop on Future Trends of Distributed Computing Systems, 2004. FTDCS 2004., 2004, pp. 80–85. doi:10.1109/FTDCS.2004.1316597.

[23] F. Kandah, B. Huber, A. Skjellum, A. Altarawneh, A blockchain-based trust management approach for connected autonomous vehicles in smart cities, in: 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), IEEE, 2019, pp. 0544–0549.

[24] S. Ruohomaa, L. Kutvonen, Trust management survey, in: International Conference on Trust Management, Springer, 2005, pp. 77–92.

[25] R. C. Mayer, J. H. Davis, F. D. Schoorman, An integrative model of organizational trust, Academy of management review 20 (3) (1995) 709–734.

[26] A. Jøsang, S. L. Presti, Analysing the relationship between risk and trust, in: International conference on trust management, Springer, 2004, pp. 135–145.

[27] Z. Lu, G. Qu, Z. Liu, A survey on recent advances in vehicular network security, trust, and privacy, IEEE Transactions on Intelligent Transportation Systems 20 (2) (2018) 760–776.

[28] W. Li, H. Song, F. Zeng, Policy-based secure and trustworthy sensing for internet of things in smart cities, IEEE Internet of Things Journal 5 (2) (2017) 716–723.

[29] A. Tajeddine, A. Kayssi, A. Chehab, A privacy-preserving trust model for vanets, in: 2010 10th IEEE International Conference on Computer and Information Technology, IEEE, 2010, pp. 832–837.

[30] J. E. Fadul, K. M. Hopkinson, T. R. Andel, C. A. Sheffield, A trust-management toolkit for smart-grid protection systems, IEEE transactions on power delivery 29 (4) (2013) 1768–1779.

[31] J. Jiang, G. Han, F. Wang, L. Shu, M. Guizani, An efficient distributed trust model for wireless sensor networks, IEEE transactions on parallel and distributed systems 26 (5) (2014) 1228–1237.

[32] R. Feng, X. Xu, X. Zhou, J. Wan, A trust evaluation algorithm for wireless sensor networks based on node behaviors and ds evidence theory, Sensors 11 (2) (2011) 1345–1360.

[33] G. Zhan, W. Shi, J. Deng, Design and implementation of tarf: A trust-aware routing framework for wsns, IEEE Transactions on dependable and secure computing 9 (2) (2011) 184–197.

[34] S. Gurung, D. Lin, A. Squicciarini, E. Bertino, Information-oriented trustworthiness evaluation in vehicular ad-hoc networks, in: International conference on network and system security, Springer, 2013, pp. 94–108.

[35] P. Golle, D. Greene, J. Staddon, Detecting and correcting malicious data in vanets, in: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks, 2004, pp. 29–37.

[36] M. D. Alshehri, F. K. Hussain, O. K. Hussain, Clustering-driven intelligent trust management methodology for the internet of things (citm-iot), Mobile networks and applications 23 (3) (2018) 419–431.

[37] B. Huber, F. Kandah, Behavioral model based trust management design for iot at scale, in: 2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics), IEEE, 2020, pp. 9–17.

[38] F. Kandah, B. Huber, A. Altarawneh, S. Medury, A. Skjellum, Blast: Blockchain-based trust management in smart cities and connected vehicles setup, in: 2019 IEEE High Performance Extreme Computing Conference (HPEC), IEEE, 2019, pp. 1–7.

[39] F. Kandah, A. Altarawneh, B. Huber, A. Skjellum, S. Medury, A human-understandable, behavior-based trust management approach for iot/cps at scale, INTERNATIONAL JOURNAL OF COMPUTERS AND THEIR APPLICATIONS (2019) 172.

[40] W. Najib, S. Sulistyo, et al., Survey on trust calculation methods in internet of things, Procedia Computer Science 161 (2019) 1300–1307.

[41] M. Nitti, R. Girau, L. Atzori, Trustworthiness management in the social internet of things, IEEE Transactions on knowledge and data engineering 26 (5) (2013) 1253–1266.

[42] R. Chen, J. Guo, F. Bao, Trust management for soa-based iot and its application to service composition, IEEE Transactions on Services Computing 9 (3) (2014) 482–495.

[43] J. Zhao, J. Huang, N. Xiong, An effective exponential-based trust and reputation evaluation system in wireless sensor networks, IEEE Access 7 (2019) 33859–33869.

[44] S. Guleng, C. Wu, X. Chen, X. Wang, T. Yoshinaga, Y. Ji, Decentralized trust evaluation in vehicular internet of things, IEEE Access 7 (2019) 15980–15988.

[45] M. D. Alshehri, F. K. Hussain, A fuzzy security protocol for trust management in the internet of things (fuzzy-iot), Computing 101 (7) (2019) 791–818.

[46] S. O. Ogundoyin, I. A. Kamil, A trust management system for fog computing services, Internet of Things 14 (2021) 100382.

[47] D. Chen, G. Chang, D. Sun, J. Li, J. Jia, X. Wang, Trm-iot: A trust management model based on fuzzy reputation for internet of things, Computer Science and Information Systems 8 (4) (2011) 1207–1228.

[48] N. U. Okafor, Y. Alghorani, D. T. Delaney, Improving data quality of low-cost iot sensors in environmental monitoring networks using data fusion and machine learning approach, ICT Express 6 (3) (2020) 220–228.

[49] L. Cui, S. Yang, F. Chen, Z. Ming, N. Lu, J. Qin, A survey on application of machine learning for internet of things, International Journal of Machine Learning and Cybernetics 9 (8) (2018) 1399–1417.

[50] H. El-Sayed, H. A. Ignatious, P. Kulkarni, S. Bouktif, Machine learning based trust management framework for vehicular networks, Vehicular Communications 25 (2020) 100256.

[51] N. Omar, H. Zen, N. N. A. A. A. Aldrin, W. Waluyo, F. Hadiatna, Accuracy and reliability of data in iot system for smart agriculture, International Journal of Integrated Engineering 12 (6) (2020) 105–116.

[52] A. Mavrogiorgou, A. Kiourtis, K. Perakis, S. Pitsios, D. Kyriazis, Iot in healthcare: achieving interoperability of high-quality data acquired by iot medical devices, Sensors 19 (9) (2019) 1978.

[53] I. Butun, S. D. Morgera, R. Sankar, A survey of intrusion detection systems in wireless sensor networks, IEEE communications surveys & tutorials

16 (1) (2013) 266–282.
[54] V. Garcia-Font, C. Garrigues, H. Rifà-Pous, A comparative study of anomaly detection techniques for smart city wireless sensor networks, sensors 16 (6) (2016) 868.
[55] H. H. Pajouh, G. Dastghaibyfard, S. Hashemi, Two-tier network anomaly detection model: a machine learning approach, Journal of Intelligent Information Systems 48 (1) (2017) 61–74.
[56] B. Ying, A. Nayak, Lightweight remote user authentication protocol for multi-server 5g networks using self-certified public key cryptography, Journal of network and computer applications 131 (2019) 66–74.
[57] J. Aas, R. Barnes, B. Case, Z. Durumeric, P. Eckersley, A. Flores-López, J. A. Halderman, J. Hoffman-Andrews, J. Kasten, E. Rescorla, et al., Let's encrypt: an automated certificate authority to encrypt the entire web, in: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, 2019, pp. 2473–2487.
[58] M. E. Glickman, A. C. Jones, Rating the chess rating system, CHANCE-BERLIN THEN NEW YORK- 12 (1999) 21–28.
[59] A. F. Agarap, Deep learning using rectified linear units (relu), arXiv preprint arXiv:1803.08375 (2018).
[60] S. Sharma, S. Sharma, A. Athaiya, Activation functions in neural networks, Towards Data Sci 6 (12) (2017) 310–316.
[61] Z. Zhang, M. Sabuncu, Generalized cross entropy loss for training deep neural networks with noisy labels, Advances in neural information processing systems 31 (2018).
[62] D. Choi, C. J. Shallue, Z. Nado, J. Lee, C. J. Maddison, G. E. Dahl, On empirical comparisons of optimizers for deep learning, arXiv preprint arXiv:1910.05446 (2019).
[63] I. K. M. Jais, A. R. Ismail, S. Q. Nisa, Adam optimization algorithm for wide and deep neural network, Knowledge Engineering and Data Science 2 (1) (2019) 41–46.
[64] Y. Zhang, G. W. Gantt, M. J. Rychlinski, R. M. Edwards, J. J. Correia, C. E. Wolf, Connected vehicle diagnostics and prognostics, concept, and initial practice, IEEE Transactions on Reliability 58 (2) (2009) 286–294.
[65] A. Ayad, H. E. Farag, A. Youssef, E. F. El-Saadany, Detection of false data injection attacks in smart grids using recurrent neural networks, in: 2018 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT), IEEE, 2018, pp. 1–5.
[66] A. L. Buczak, E. Guven, A survey of data mining and machine learning methods for cyber security intrusion detection, IEEE Communications surveys & tutorials 18 (2) (2015) 1153–1176.
[67] M. E. Glickman, The glicko system, Boston University 16 (1995) 16–17.
[68] S. Jarvenpaa, T. R. Shaw, D. Sandy Staples, Toward contextualized theories of trust: The role of trust in global virtual teams, Information Systems Research 15 (2004) 250–267. doi:10.1287/isre.1040.0028.
[69] S. Saxena, I. K. Isukapati, S. F. Smith, J. M. Dolan, Multiagent sensor fusion for connected & autonomous vehicles to enhance navigation safety, in: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, 2019, pp. 2490–2495.
[70] D. Krajzewicz, J. Erdmann, M. Behrisch, L. Bieker, Recent development and applications of sumo-simulation of urban mobility, International journal on advances in systems and measurements 5 (3&4) (2012).
[71] D. Krajzewicz, Traffic simulation with sumo–simulation of urban mobility, in: Fundamentals of traffic simulation, Springer, 2010, pp. 269–293.
[72] M. Haklay, P. Weber, Openstreetmap: User-generated street maps, IEEE Pervasive Computing 7 (4) (2008) 12–18.
[73] D. M. Gavrila, V. Philomin, Real-time object detection for" smart" vehicles, in: Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol. 1, IEEE, 1999, pp. 87–93.
[74] G. S. Khekare, A. V. Sakhare, A smart city framework for intelligent traffic system using vanet, in: 2013 International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), IEEE, 2013, pp. 302–305.