Journal of
Sensor and Actuator Networks

MDPI

Review

Sensor Network Environments: A Review of the Attacks and Trust Management Models for Securing Them

Kealan Mannix [1,2,*], Aengus Gorey [3], Donna O'Shea [2,4] and Thomas Newe [1,4,*]

[1] Department of Electronic and Computer Engineering, University of Limerick, V94 T9PX Limerick, Ireland
[2] Confirm-Smart Manufacturing Research Centre, University of Limerick Digital District, V94 C928 Limerick, Ireland
[3] Digital Platforms and Security Platforms, Analog Devices International, V94 RT99 Limerick, Ireland
[4] Department of Computer Science, Munster Technological University, T12 P928 Cork, Ireland
* Correspondence: 17239664@studentmail.ul.ie (K.M.); thomas.newe@ul.ie (T.N.)

Abstract: Over the past decade, new technologies have driven the rise of what is being termed as the fourth industrial revolution. The introduction of this new revolution is amalgamating the cyber and physical worlds, bringing with it many benefits, such as the advent of industry 4.0, the internet of things, cloud technologies and smart homes and cities. These new and exciting areas are poised to have significant advantages for society; they can increase the efficiency of many systems and increase the quality of life of people. However, these emerging technologies can potentially have downsides, if used incorrectly or maliciously by bad entities. The rise of the widespread use of sensor networks to allow the mentioned systems to function has brought with it many security vulnerabilities that conventional "hard security" measures cannot mitigate. It is for this reason that a new "soft security" approach is being taken in conjunction with the conventional security means. Trust models offer an efficient way of mitigating the threats posed by malicious entities in networks that conventional security methods may not be able to combat. This paper discusses the general structure of a trust model, the environments they are used in and the attack types they are used to defend against. The work aims to provide a comprehensive review of the wide assortment of trust parameters and methods used in trust models. The work discusses which environments and network types each of these parameters and calculation methods would be suited to. Finally, a design study is provided to demonstrate how a trust model design will differ between two different industry 4.0 networks.

Keywords: trust; trust model; security; networks

Citation: Mannix, K.; Gorey, A.; O'Shea, D.; Newe, T. Sensor Network Environments: A Review of the Attacks and Trust Management Models for Securing Them. *J. Sens. Actuator Netw.* **2022**, *11*, 43. https://doi.org/10.3390/jsan11030043

Academic Editor: Jordi Mongay Batalla

Received: 30 June 2022
Accepted: 4 August 2022
Published: 8 August 2022

1. Introduction

Trust models are increasingly being used in a wide range of sensor networks, such as the IoT (internet of things), VANETs (vehicular ad hoc networks), cyber-physical manufacturing and a wide range of wireless sensor networks. They are not just limited to the sensor network field, but are also applicable to many network topologies, such as online P2P (peer too peer) networks [1] and web services [2]. However, this paper focuses mainly on sensor and IoT networks. The reason for the necessity of this new security measure is the fact that these networks are vulnerable to internal attacks, such as bad-mouthing (an attack where a malicious node falsely reports indirect trust values of another node), collusion, on-off and Sybil, to name just a few. These attacks can pose serious threats to the network operation if not dealt with in a timely manner, which conventional security is not capable of doing.

Conventional security measures such as encryption, digital signatures and authentication measures (such as passwords, tokens or biometrics) are all aimed at protecting the "CIA triad (modified)" (Confidentiality, Integrity and Authenticity (modified triad as in the classical CIA Triad; this is Availability)). The concept of the triad security is aimed at protecting from external attackers, who should not have access to the system or network. However, many of these new network topologies, such as VANETs and IoT

networks, are operating on the basis that nodes may enter and leave the network with relative ease, allowing the nodes to send and receive data within the network. In the case of WSNs (wireless sensor networks), many of the nodes will not have high computing capabilities and can be quite vulnerable to cyber-attacks. Due to the potential openness of their deployed environment and transmission medium, the nodes could also be vulnerable to physical attacks, such as tampering or hijacking attacks [3]. All these attacks are internal to the network and cannot be conveniently mitigated by conventional means of security, such as cryptography and authentication.

Trust models are a simple concept used to protect against the aforementioned internal attacks. Trust models are simply a method of interpreting the data provided by nodes in the network to determine if they are trustworthy or not. This concept is similar to, and based on, how people establish trust in their everyday lives. Humans are able to interpret large amounts of data from the world around them and use these data to make decisions, such as whether a person be trusted. For example, if you met a stranger whom you just witnessed stealing something from a shop, you would be much less likely to trust something they said than if you met a friend you had known for years, and who has never, to your knowledge, done anything bad. It is from these simple concepts that trust can also be developed in networks. Nodes that have had good previous interactions with one another will afford one another a higher level of trust than they will afford a new "stranger" node that appears to be sending peculiar data in the network.

The data used to calculate trust can be divided into three main categories: direct trust, indirect trust and physical data. These categories are also what people use in their assessment of trust, and can be thought of in a synonymous way when applied to the cyber world. The three categories are explained very briefly below, along with analogies of how people tend to use them in their trust calculations of another person.

Direct trust: Your own personal, direct experience with the entity—are they a friend you have known for a while?

Indirect trust: Another entity's recommendation of the entity—do you have a mutual friend? What are their thoughts on them?

Physical data: Observations made of the entity's behaviors or attributes—have you witnessed them misbehaving? What is their demeanor like?

The basic concept of a trust model is simple and familiar; however, there are many different ways of implementing a model and many contrasting use cases in which trust models can be used. This paper aims to provide a comprehensive survey of the various trust models used in sensor networks and to provide an outline of how these models work, along with their strengths and weaknesses. Other reviews on trust models also provide this information; however, they lack an analysis of the measurement parameters of trust. Other works also do not provide a view on how the environment and network types can affect the design and operation of the trust model. This work aims to contribute to these lacking areas by discussing the parameters seen to measure trust in many models and conducting a design study of a trust model on two different industry 4.0-type networks.

The rest of the paper is outlined as follows. Section 2 discusses some of the previous survey papers produced in the area of trust models. Section 3 outlines the requirements, structure and theory that a trust model should follow. Section 4 outlines in detail the types of attacks that are currently being mitigated by trust models, and Section 5 describes the parameters that can be used in the calculation of trust. In Section 6, a detailed look at how trust is mathematically calculated in the assorted models is presented. In Section 7, some recommendations for the design of new trust models and potential new research areas are considered. Section 8 considers how different environments need different methods for calculating trust by comparing a Factory 4.0 to an Agriculture 4.0 environment, before concluding the work in Section 9.

2. Previous Works

There have been previous reviews carried out on the topic of trust management and modeling. A detailed comprehensive study of trust modeling is conducted in [4], where a detailed look into what trust actually is, not only from a digital perspective but also from a human social perspective, is conducted. Many attack types and mathematical trust calculation techniques are discussed, along with recommendations for what a trust model should provide in terms of security.

In [5,6], a brief overview of trust models in VANETs is provided. Both papers offer a perspective on the characteristics of VANETs and their unique security challenges, as well as the attack types they are facing and the models that have been used to attempt to mitigate these attacks.

In [7], a specific look at fuzzy trust modeling techniques in cloud computing environments is offered, providing a detailed overview of the cloud infrastructure, along with trust models being used to secure this fast-growing technology.

A discussion of the challenges of traditional security measures in [8] considers how trust is used as a method for overcoming these challenges. The use of trust models for providing authentication, access control, secure service provisioning and secure routing is also analyzed in this paper, along with the usual run-down of the attack types and robustness of trust models.

The previous works all contain an analysis of the attack types and model types. This paper aims to provide an understanding of the attack and model types, but also aims to provide an understanding as to how trust models are generally structured and used. From reviewing previous works in the area, there appears to be a gap in the research regarding the actual parameters used to measure trust. This paper takes a much closer look at the specific parameters used to provide the data for trust calculations (Section 5). The parameters used in the measuring and calculation of trust can be just as, if not more, important to the model than the mathematical classification technique itself.

3. Trust Model Concepts

Trust models differ vastly from model to model depending on many factors, such as environment, use cases and the level of security needed. However, many key concepts remain the same for all models. In this section, a general overview of the trust models is discussed, including the network types they are being used in, the security features they aim to provide and why these features cannot be provided by conventional techniques. Lastly, the general structure of a trust model is considered.

3.1. Environments

While trust models can be deployed in a number of areas, this paper focuses on models used in sensor networks or similar network types, where the nodes of the network are communicating data back to a central system. There are many types of network topologies and attributes. Featured below is a list of some of the main ones, with a brief definition of each. Networks are not limited to just one of these attributes and are often a combination of a few. For example, you could have a heterogeneous, static, clustered network.

- Homogeneous: a network where all nodes are identical—same function, device, rank, etc.
- Heterogeneous: a network where nodes are not all identical.
- Hierarchical/clustered: networks where some nodes have a higher rank than others. Often, the network is controlled by a base station, which is in command of lower-level nodes or in a clustered network cluster head. These cluster heads are each in command of their own cluster of edge nodes. It is usually the case that the higher ranked the node is, the more computing power and resources it has.
- Static: a network where nodes are neither leaving nor entering—e.g., a smart factory sensor network usually has the same sensors all the time. Note: sometimes static can

be referred to in terms of the nodes' physical positioning. In this paper, static refers to the above definition unless explicitly stated otherwise.

- Dynamic: a network where nodes are entering and leaving the network, e.g., a mobile network or VANETs. Note: sometimes dynamic can be referred to in terms of the nodes' physical positioning. In this paper, dynamic refers to the above definition unless explicitly stated otherwise.

Each network type will have different properties and challenges and will require slightly contrasting methods of determining the trust of its nodes. What follows is a description of some of the common types of networks that trust models are currently being used in.

IoT/CPS: Internet of things and cyber-physical systems are similar in terms of their network structures. They are a rapidly advancing area, which can be applied to many different use cases, such as smart homes or cities, smart medical equipment/monitors and industry 4.0. Internet of things networks are, in general, a heterogeneous network comprised of many different sensors and controllable devices, all communicating with each other across a network. Trust models are applied to IoT-type networks in [9–17]. The challenges IoT networks tend to face are very common among most of the network environments. These challenges include potentially low computational power and low power resources (if battery-operated) for the network nodes, as well as the deployment of network nodes in a wide array of areas, which can leave some nodes vulnerable to physical compromise. Nodes within the network can be corrupted due to cyber or physical means, introducing internal attack threats to the network which cannot be effectively diminished by conventional security.

Mobile crowd sensing: [18,19] A subset of IoT applications, mobile crowd sensing is where large groups are used to collect data on some type of metric, such as traffic or noise. Many mobile phone apps use crowd sensing to collect data, such as google maps, which collects data on your location and speed to provide information to other users on traffic flow and journey times. Crowd-sensing applications use trust models to determine whether the data they get are trustworthy and useful or not, as in [19].

Wireless sensor networks: WSNs [3,12,13,20–29] are perhaps one of the most common use cases for trust models. This is due to the fact that sensor nodes, more often than not, have very limited computing capabilities. Trust models can be developed in very lightweight manners to account for this. Their wide range of deployment and low computing capabilities make them vulnerable to being compromised in a similar manner to the previously mentioned IoT devices. Sensor networks often have a central node. Trust calculations for the sensor nodes are often performed in this central node or base station, which has more power and resources available to it. This type of topology is well-suited to a trust model, and some very good results can be achieved by introducing a trust model to the network. There are many sub-types of WSNs, including homogeneous and heterogeneous, underwater acoustic networks [23,28], large scale networks [29], hierarchical networks [12] and cluster-based networks [3,13]. Each type of network will have slightly different requirements and topologies. Specific trust models can sometimes be used in more than one specific type of WSN, if they incorporate features that allow for flexibility in their implementation (more on these features is provided in Section 6).

VANETs: Vehicular ad hoc networks [30–32] are comprised of vehicles, such as cars, trucks, buses, etc., which accept and receive data from one another and from roadside units (RSUs). They are used in smart cities for applications such as traffic monitoring and optimizing, accident prevention and early warning systems. These networks are much more dynamic, with vehicle nodes moving in and out of the network freely and the nodes not having a fixed point, which, of course, brings with it potentially malicious nodes entering and leaving freely. With these dynamic networks, some unique aspects can be added to their trust models, such as distance-checking as a plausibility check on the data [30,31], among others.

Unmanned aerial systems: This area is quite similar to a VANET. A trust model is applied to one in [33]. These systems can be thought of a cross between a VANET and IoT-type network, with the nodes, in this case, being static in the network but dynamic in terms of their positions, as they are traveling.

Cyber-physical manufacturing: [34,35] This type of network can be defined as a specific use case of the IoT or WSNs and is comprised of sensors and devices used to control critical infrastructure in factories. Often, these networks are more secure from the physical attacks of nodes, with only authorized personnel allowed enter the factory/manufacturing area and no new nodes allowed to enter the network without authorization. However, the nodes are still at risk of being compromised, and trust models offer an effective lightweight solution suitable to this environment.

3.2. Security Features

Conventional security focuses on providing hard security to a system based on the CIA triad (modified)—Confidentiality, Integrity, and Authenticity (as well as availability and access control). The trust model aims to provide most of these security features both internally and externally on the network.

Conventional security techniques include methods such as encryption, digital signatures, digital certificates and authentication methods (passwords, tokens, biometrics). All these methods mostly assume and are tailored according to the premise that the attacker is an outside entity, as is the case in attacks such as eavesdropping, DDoS or man-in-the-middle (MitM)-type attacks. These methods establish connections between users and assume trust thereafter for the duration of the session. However, in the new network environment types, this is often not good enough, and a new 'zero trust' approach must be taken, as nodes within the network can be compromised or gain access legitimately before launching their attacks, effectively launching them from inside the conventional security barriers.

Conventional security methods can still be used within the trust models to provide services such as confidentiality, integrity and authentication, although they must often be adapted to suit low-powered nodes. This is conducted through lightweight or edge cryptography [36,37] and key management [37] methods.

Trust models use the concept of zero trust, in which no entity is trusted until verified. The trust is dynamically updated. This is usually achieved through time-triggered updating or event-triggered updating. This dynamic updating aims to prevent nodes or entities that were once trusted but then became malicious from launching attacks internally on the network.

The main security contribution of a trust model is a form of access control. All trust models should be able to provide and deny access to any node on the network. Depending on the use case scenario, models will decide whether a node can be trusted, whether the data can be trusted and used, or whether the node should be allowed to access some form of resources on the network. Models must be able to discard any untrusted data and eject any untrusted or malicious nodes. This access control provided by the models will successfully mitigate attacks, such as packet modification, bogus data injection or malicious software spreading, which can be carried out through a number of different initial attack methods.

Trust models can also provide authentication if the situation requires. This can be achieved by means of signatures, as seen in [12], where an intrusion detection service provides a signature-based authentication method. As seen in [9], a hardware approach can be taken for the authentication of nodes with the use of radio frequency distinct native attribute (RF-DNA) fingerprinting. This method analyzes the transmitter signal of the node, comparing it to known samples from previous communications. In [9], these data are classified using an SVM (support vector machine). This method is similar to human biometrics (as if the transmitter of the device is the biometric) and definitely has great potential to be used in static hierarchical or clustered network environments, where there

is a cluster head or base station with the capabilities of running the required hardware and software.

3.3. Trust Model Structure

For the purposes of this analysis of trust models, a high-level block diagram structure of a trust model was contrived, as seen in Figure 1. It is important to note that this is not based on any particular model and by no means defines all trust models. Many models loosely follow this structure, potentially leaving out or adding in components to the structure to suit their specific applications. This high-level block diagram includes the most common and pivotal features of trust models and aims to provide an understanding as to how a trust model typically operates. What follows is a description of each section numbered in the diagram.

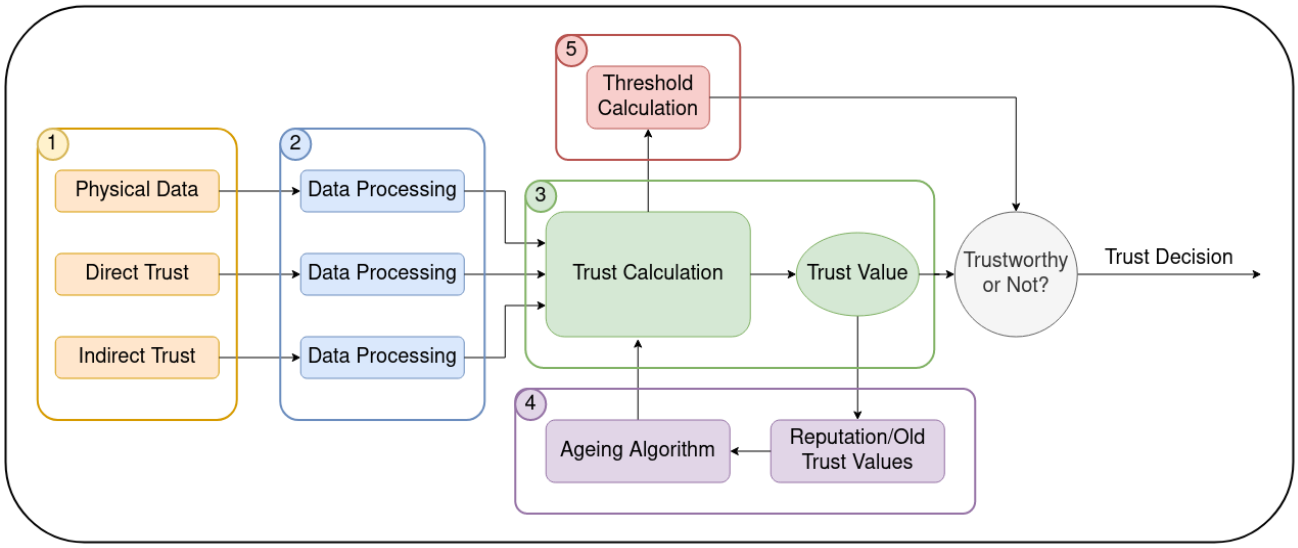

Figure 1. Diagram of a trust model's basic structure.

(1) Data gathering. The data gathering phase is essential to every trust model, as there must be data in order to calculate trust. The data gathering phase mainly collects data from three groups (just one or a combination of multiple groups can be used). These groups are shown in Figure 1.

Physical data include data such as energy and resource consumption of the device, the device status (software/OS version) and packet characteristics. Essentially, this includes any physical data about the device. These data can be used to detect odd behavior or anomalies in the device and can also be further used to determine direct and indirect trust.

Direct trust refers to data obtained from interactions between the trustee and trustor. This type of data can be obtained through monitoring the communication behavior of, or the packets being sent by, a device, or through other physical data that is obtained, or by another method of defining whether the interaction was successful or not. It is often determined through probabilistic means, based on the communication behavior of the device.

Indirect or neighbor trust is essentially direct trust obtained from another source. For example, if node A wishes to communicate with node B, node B can calculate its own direct trust with A, but also ask nodes C and D for the direct trust values between themselves and A. B can then take this value into account when calculating its trust of A. Usually, the indirect trust is accumulated from all nearby neighbors and summed or averaged in some of them, and this is taken into account with less weight of effect than the direct trust values.

(2) Data processing. In some models, the data are pre-processed before being used in the trust calculation. In these types of models, the data are pre-processed using a mathematical model individually, and the trust calculation acts as a form of integrating

these values together, such as a weighted average or sum. This is not a necessary step; however, it can be useful in some instances. For example, direct trust can be calculated using physical data or potentially by considering previous trust values or more than one parameter. Multiple indirect trust values are obtained from neighbors and need to be amalgamated. These calculations can take place before the trust calculation, if necessary, depending on the model being used.

(3) Trust calculation. This phase is at the heart of the trust model. It is where all the data obtained are converted to a meaningful trust value to determine whether the device/node is malicious or not. This calculation can be performed in numerous ways, using different mathematical models. These models are discussed in further detail in Section 6. From the block diagram in Figure 1, we can see that all the data gathered are used in this phase, and previous trust values can also be fed back and used in this phase. The trust value that is calculated in this phase is updated regularly, and this can be accomplished through time- or event-triggered updating, depending on the model.

(4) Historical trust values. Many trust models, but not all models, take into account historical trust values when calculating their trust. These previous values are stored and inserted into the trust calculation, usually using an ageing algorithm that can be anything from a weighted average to something more complex. These historic trust values are often referred to as the reputation of the node. This step may not be utilized in all models, and the direction arrows in Figure 1 show this.

(5) Threshold calculation. An area that is often lacking in most models is dynamic threshold calculation. The threshold level in most models decides whether a node's trust value renders it malicious or not. Dynamically calculating the threshold gives the model more flexibility to be applied in more than one specific network, as it will account for differences in network traffic, the number of nodes or other factors that could affect the average value of trust for a node. The majority of models tend to leave this phase out and instead employ a static threshold value method [38]. Dynamic threshold calculation should, however, be considered for the models, as it provides more flexibility in allowing them to be applied in more than a single specific network.

The final stage in the diagram is the simple comparison of the trust value with the threshold value. This stage determines the final decision on the node's trust status. If a node becomes ejected from the network, some models have methods that allow these nodes to build up reputation again over time without having full access to the network and, later, to potentially rejoin the network. Models also sometimes face the 'cold start' issue, whereby trust calculations that rely on data and previous values do not work initially on startup, which must be considered. These points are discussed further below.

4. Attack Types

There are numerous attack types that can be carried out against sensor or IoT networks. Many of these attacks are often launched individually, but they can be used in conjunction with one another. Some attacks are designed specifically to gain access to the network or boost (or diminish) the trust of a given node, while others are designed to spread malware, disrupt routing and drop the critical information in transit. What follows is an overview of the attack types that are typically found in the network-type environments and that trust models are being used to mitigate. The attacks have been classified into six broad categories: access, reputation, payload/active, denial of service (DoS), routing and physical. Often, the access- or reputation-type (and sometimes the physical-type) attacks are used to gain privileges in the network that allow the payload, DoS or routing attacks to be run, which can in turn cause network disruption. A lot of these attack types are carried out by compromised nodes, which can become so through access attacks or device hacking. A review of the methods by which nodes are compromised is beyond the scope of this paper.

4.1. Access Attacks

Counterfeiting. Also known as impersonation, spoofing or identity fraud attacks, counterfeiting involves inserting a malicious node into a network that will pretend to be a legitimate node that is already in the network. It can be carried out using a new node with a fake ID that is inserted into the network, or by changing the ID of a compromised node already in the network to make it appear as a different node. This can be achieved when a relaying node steals the ID of a node from relayed data packets and uses this on its own packets [39]. These attacks can be used to gain access to networks and launch further attacks and, in some cases, divert the mitigation actions of the trust model back to the node that is being counterfeited, causing the legitimate node to be ejected. The authors of [39] propose a detection method for impersonation attacks based on bloom filters and the use of hashes for WSNs.

Man-in-the-middle (MitM). One of the more common cyber-attacks in all areas, the MitM attack involves an attacker injecting itself into an ongoing communication between two legitimate parties [40]. The attacker becomes an intermediary in the communication, whereby both legitimate parties still believe they are communicating directly with one another, but, in fact, they are unknowingly communicating with one another through the MitM. MitM attacks can be passive in nature, simply eavesdropping or running a traffic analysis, although often they are used to launch further attacks, such as those in the payload or DoS categories.

4.2. Reputation Attacks

These attacks specifically affect trust models. They are quite similar in terms of the goals they wish to achieve. Different reputation-type attacks are often used interchangeably or in conjunction with one another to attack vulnerabilities in a given trust model.

Breakout fraud/on-off. Sometimes referred to as a zigzag [30] or garnished attack [22,29], this type of attack involves a node behaving as expected by the network for a period of time. Then, when its trust values are high enough, it will launch an attack. When the model detects suspicious behavior from the node and the trust value begins to fall, the node can stop behaving maliciously and return to normal behavior, which in turn will restore the trust value before the network ejection is initiated. This process can be repeated multiple times, as long as the node does not let its trust value drop below the threshold value. Attacks such as this demonstrate the need for a dynamic and fast trust-updating response to catch attacks early, before they disappear again for a period of time.

Bad-mouthing. Bad-mouthing involves a malicious node falsely reporting the indirect trust values of another node. This will negatively affect its trust value. Collusion or sybil attacks can be used to boost the effectiveness of this attack. Some models use methods where only positive trust values can be reported back to counteract bad-mouthing attacks [20].

Ballot-stuffing. As it is in the physical world, ballot-stuffing involves one user stuffing the ballot with the same vote to change the results of a vote. This scenario is the same in the digital world, where a node can stuff the ballot in a majority voting system or an indirect trust system. This can affect its own or other nodes' trust values. This attack can be achieved through means of sybil, or collusion, and this technique can also be used for self-promoting and bad-mouthing attacks in models that are vulnerable to it.

Collusion. This attack can be considered as a reputation-based attack or a routing attack (or it can also be used for forms of DDoS attacks). In this attack type, multiple attackers work in collusion to modify packets, drop routing packets [41], perform bad-mouthing or self-promoting attacks or compromise majority voting schemes. It is a very broad class of attack that essentially describes any attack where there are multiple attack nodes working together to achieve a common goal.

Sybil. In this scenario, a node presents multiple identities to other nodes in the network [42]. These identities can be forged or stolen from other nodes. This allows the node to have a greater say in the network and can be used to boost bad-mouthing or

self-promoting attacks. It can also be used to corrupt data, majority voting schemes and other processes in the network.

Self-promoting. This is an attack where the node simply reports better trust values for itself or another node that it is working with. This attack aims to boost the trust value of the attacker, giving it more access to launch further attacks. This attack can be launched using sybil or collusion attacks, as mentioned above.

4.3. Payload/Active Attacks

These types of attacks are launched from compromised nodes after access is gained and trust is established. These types of attacks can be very destructive to a network and the goal of the trust models would be to detect and eject compromised nodes before they can gain enough power to launch such attacks effectively.

Malicious software spreading. This is the spreading of software such as worms or viruses through a network from a compromised node. These worms and viruses can compromise more nodes and propagate throughout the network, affecting the operation of compromised nodes in any way that the attacker intended when creating the malware. These attacks can be devastating to a network.

Packet modification. In this scenario, a compromised node modifies all or some of the packets that it is supposed to forward [43]. This allows the attacker to insert the data it wants into the packets, such as false data readings or malware. Data pollution attacks are also very similar to packet modification and involve the injection of corrupted data into the network to disrupt normal operation.

Selective forwarding. Otherwise known as packet dropping, this is where a compromised node drops all or some of the packets that it is supposed to forward [43]. This causes data to be delayed in transit or lost completely. Nodes can also store packets and forward them later in a replay attack. A replay attack can be mitigated using random numbers or timestamps.

4.4. Denial of Service Attacks (DoS)

Denial of service. DoS attacks are attacks that intend to shut down a machine or network, making it inaccessible for its intended users [44]. They are accomplished by flooding a target or sending it information, or by programs which cause it to use up more resources (CPU, memory, energy) than it should and eventually crash. Most of the attacks in the routing section can also be classed as DoS attacks, as they deny the service of communications to the intended users.

Distributed denial of service. A DDoS attack is a DoS attack carried out by multiple entities. The intentions of a DDoS attack are the same as those of a DoS attack, but distributing the attack creates more attack power (due to more machines), making it harder for the target to combat. It also creates more than one attack point, making it more difficult to detect the attack location and block the malicious entities.

Energy exhaustion. This is a type of DoS attack that can be especially concerning for low-power or wireless networks. In some networks, the nodes may operate on battery power, and if an energy exhaustion attack is successfully carried out over a short period of time, the nodes can run out of power. Some of these nodes are in hard-to-reach areas, costing the organization that owns the network money and time by creating a need to replace the batteries early. A good example of this is provided in [23,28], where the nodes are in underwater networks, making node access difficult and energy exhaustion attacks detrimental to the networks' operations, which could cause the nodes to be down for a prolonged period of time due to a dead battery.

4.5. Routing Attacks

In most cases reviewed in this paper, the networks are a type of ad hoc network, meaning that there are not typical routers in the network. However, all nodes, or a portion of nodes (depending on the topology), essentially act as routers within the network and

can be targeted for such routing attacks. The authors of [41] provide an overview of some common routing protocols and routing attack types in ad hoc networks, and the solutions to these using routing protocols. These attacks can also be thwarted through the use of trust models in many cases, hence they are mentioned here. Ideally, a good routing protocol and trust model would be used to ensure the most secure approach to these attacks; however, routing protocols are beyond the scope of this paper and, therefore, they are not discussed here.

Hello flood. This attack uses Hello packets to announce to many nodes that they are neighbors. A laptop-class adversary can be used to send this packet to all the nodes in a network to convince them that a compromised node is their neighbor [42]. This type of attack can be used to disrupt routing or, in clustered network structures, to fool the cluster head selection process into choosing the compromised node as a cluster head, placing it in an ideal position to launch further attacks.

Wormhole. This attack requires two or more attacker nodes. One attacker receives packets and 'tunnel' them to the other through a private high-speed network. This allows the attackers to deliver packets with better metrics than a normal multi-hop route [45]. This allows the attacking nodes to obtain more traffic, as they provide the fastest route which in turn makes them a threat for launching further routing, payload or DoS attacks on the network. Wormhole attacks can also be used to secretly 'tunnel' information out of a network.

Black-hole. [46] A black-hole attack is one in which the attacker node simply drops all packets sent to it (just like a real black-hole, where things can only fall in but never come back out). This prevents the packets from reaching their target destination and, in situations where routing replies or packet acknowledgments are required, causes errors in the network and becomes very disruptive. The upside to black-hole attacks is that they are relatively easy to detect.

Gray-hole. Gray-hole attacks are similar to black-holes, but instead of discarding all the information sent into it, the gray-hole attack only discards some packets or parts of packets. This allows traffic to continue flowing and replies and acknowledgments to be received, but with a subtle loss of information, making this type of attack much harder to detect than a black hole.

Sinkhole. This is an attack where a compromised node tries to attract all the traffic from neighbor nodes based on the routing metric used in the routing protocol [47]. With almost all the network traffic passing through this node, it can launch further attacks, such as packet modification, or black-hole-, gray-hole- or reputation-type attacks.

4.6. Physical Attacks

These types of attacks are not within the scope of a trust model, in the sense that they cannot be detected or mitigated by them. A brief examination of these attacks is offered here, as these attacks can be carried out easily on sensor or IoT networks and can provide a means of compromising nodes or taking them offline, which the trust model must then in turn be able to handle.

Device breaking. A simple attack that should be considered when setting up sensor network environments, device breaking is as simple as smashing a physical device. This will, of course, knock the device off the network, so models should be prepared that are resilient in the case of one or more devices dropping off the network.

Device tampering. Tampering can be used as a way of gaining easy access to a node through physically plugging a device (such as a laptop) into a node or modifying something on the hardware (swapping out memory cards, for example). There are many different ways of physically tampering with a device that will result in its compromise, giving attackers access to the network through the device or allowing them to insert worms or viruses into the network.

Jamming. This type of attack makes use of intentional radio frequencies to harm wireless communication systems by keeping the communicating medium busy [48]. This

physical attack essentially blocks legitimate nodes from communicating by overpowering their signals. The attack can be executed in many ways, such as by simply emitting the frequencies used, by sending random packets or by more advanced methods that can evade detection for longer. The authors of [48] examine the various methods of jamming and their detection and countermeasures. Techniques such as link quality indicators (LQI) similar to those used in [25] could potentially be used to ensure that the significantly reduced communications quality caused by jamming will not greatly affect the trust values of the network.

5. Trust Data Gathering

The information in this section relates to phases one and two of Figure 1. This section analyzes the methods of gathering data and the parameters measured by the models to be used in trust calculation.

5.1. Data Gathering

Direct gathering. Almost all models use some form of direct trust. Direct trust is any trust metric gathered directly between trustor and trustee, without the use of a third-party node or entity. This can include analyzing the data sent in the communications, i.e., packets, communication frequency and the quality of the data [19], or using a system similar to a watchdog mechanism. In [20], a watchdog mechanism is used to monitor the routing, processing and data in nodes that are in direct communication with a given node. Many trust parameters can be gathered directly through this direct gathering and used in trust calculations later on, or to calculate a direct trust score.

Indirect gathering. The opposite of direct gathering, indirect gathering is the method of using third-party nodes or entities to gather data and use these data to calculate indirect trust scores (also referred to as second-hand, recommended or reputation trust scores). This is a very common method of gathering data. Methods for selecting the nodes to gather this data include one-hop neighbors [16,20,38], nodes in the cluster [13,14,24,29,31], cluster head or base station feedback [22], multi-hop neighbors [26] and common neighbors between the trustor and trustee [25].

Broadcasting. This is a method in which a node or cluster head broadcasts information. In [21], all communications are broadcasted from the beacon nodes so that other beacons can listen and check if the location information is correct when compared to their location table. In [35], the gateway information of a provider is broadcast and opened up for auction to the users, and, in [38], trust tables are multi-cast (it is multicast as it is not broadcasting to the entire network, but only its cluster) from the cluster head to its cluster after a table update has taken place. The broadcasting method removes a layer of privacy from the data being broadcast and prevents attacking nodes from fooling individual nodes, as all the nodes are able to check if a malicious node broadcasts inaccurate information relative to the other nodes in the group.

5.2. Trust Parameters

Communications are a vital part of any network and many attacks. Attacks can often change communication behaviors in a network; therefore, it makes sense to use communications as a parameter for measuring trust. Many models use some attribute of the communications between nodes to help to model trust. The message frequency and packet forwarding rate [9,17,32], as well as the packet forwarding behavior [24], are simple ways of determining if a node is behaving unusually. Another common approach is to analyze the ratio of successful interactions/packets sent, or that of unsuccessful or total interactions/packets sent. This type of communications trust is used widely (reference [3] terms it as honesty trust) [14,22,23,25,27,29,31]. This approach requires a method of defining a successful or unsuccessful transmission, which is usually achieved using a simple set of rules defined for the given context. Each model will differ in how it defines successful or unsuccessful transmissions, but a common approach is to use an ACK signal to indicate

the received packets. The communication trust score is calculated using simple ratios, equations or probabilities to convert the data collected into a value that can later be used in the trust calculation. The method used to achieve this is slightly different for each model so as to suit its contextual needs.

Data can be attacked through some of the attack methods mentioned in Section 4, and these attacks can modify and change data being sent by nodes in a detectable manner. Data checking can often be performed in simple yet effective manners, adding little computational overhead to the network. The authors of [3,20,24,25,28,49] use simple anomaly-type checks, such as comparing sensor data to the average value of the data from all sensors. If the data is flawed, then there may be a potential compromise. The authors of [30,31] use plausibility checks on the data, both operating on a VANET network and employing methods for determining vehicles' distance from the roadside unit that they are in communications with. If the data indicates that it is being sent from a location outside this range, these checks will catch it and instantly discard the information. These types of checks seem trivial, but they are very effective and add little computational overhead to a model and, therefore, it makes sense to include them.

Other models use methods such as analyzing the expected values of the amount of data being collected [17], analyzing temporal [15] or spatial [23] correlations of the sensor data, and detecting if a packet is malicious based on its signature [12] or successful data interactions [29].

In [19], a quality of data measurement is used to develop a QoD score for nodes in a crowd-sensing network, and this value influences the trust value of the node. There are eight quality dimensions used to attain this QoD score. Six of these data qualities are defined in [50] as accuracy, completeness, reliability, consistency, uniqueness and currency. The remaining two are defined as syntactic accuracy and semantic accuracy, in the case of live sensor data. This method of QoD assessment works well for applications such as crowd sensing but could also be applicable to other applications, such as some IoT or WSNs.

Interaction history. Often, nodes will have had previous interactions with each other. Just as people develop (or lose) trust in each other through more interactions, nodes in a network can too. This can be achieved through counting the number of previous successful or unsuccessful interactions between nodes, and by allowing nodes to be more trusting with entities they have a good history with and more cautious with entities they have bad or no history with. The authors of [3,10,31,32] all consider the number of previous interactions that nodes had with each other. This method is similar and overlaps with the communication method discussed above, which analyzes the ratio of successful to unsuccessful/total packets sent to a node, which is also using information about past interactions to help to measure trust.

Resource consumption can be a clear indicator of a DoS, routing, payload or other type of attack. If a node is consuming an abnormal amount of energy, CPU power or bandwidth, then it is most likely doing something it should not be, such as running CPU intensive processes, downloading programs or running far more communications than usual. These are all telltale signs of many of the attacks discussed in Section 4. Numerous models in the aforementioned communications section can detect increased bandwidth usage by monitoring the communications.

CPU power is monitored in [20] using a watchdog mechanism that allows nodes to observe their neighbors' processing consumption. CPU, network and storage trust is used to help to define trust for service composition in mobile cloud environments in [18] and in the IoT domain in [10].

The authors of [23,25,27,28] all use an energy trust value based on the energy consumption rate of the node. It is important to note that, in these networks (WSNs and underwater WSNs), energy consumption is very important, as the sensors are in difficult-to-access, remote locations with limited battery life, and excessive energy usage due to attacks or malfunction can cause the sensors to be out of action for a prolonged period of time. Therefore,

it makes sense for networks such as these to use energy usage as a trust-measuring metric, as energy is such a valuable resource for them.

Device behavior is examined in [10]. The behavior concerns the device resource usage metrics and whether they are within a predetermined range for the device. This information is obtained from MUD (manufacturer usage description) files. MUD files consist of an MUD URL, where the network can find information about the device connecting to it from the MUD file server [51]. This information notifies the network about what this physical device does, and the network can take appropriate access control measures and can get an accurate idea of its normal resource usage for anomaly detection.

Device status encompasses information regarding device integrity and device resilience [10]. The integrity of the device is defined in [10] as the extent to which the device is known to run legitimate firmware/OS/software in valid configurations. The resilience of the device is defined as the known security of its firmware/OS/software versions. The authors of [10] apply this metric to the trust calculation of the device. The device most likely will not change its firmware/OS very frequently. Therefore, this metric may be more useful in helping to calculate an initial trust value for a new node joining a network when there are little to no other available data on the node.

Associated risk is not commonly used in trust models. It is used in [10], where the risk associated with a device is classed as a fuzzy level. The risk is calculated based on the device's probability of being compromised and the damage it could do to the network (based on its access rights and perceived value and also its neighbor's criticality and perceived value in the network). This method involves a great deal of work in classifying the nodes' associated risk accurately, and it may not be useful in many network types (e.g., homogeneous networks). However, in more complex networks, it may be useful for implementing different classes of trust for devices with different priorities and capabilities within the network in order to increase productivity. A similar concept is used in [30], where a role-orientated trust system is used. In the VANET, different weights are given to the information received from normal vehicles, high-authority vehicles (emergency vehicles), public transport, professional vehicles and traditional vehicles (vehicles with little to no travel history, which are assigned a lower weight).

Task success may not be very applicable in most network environments, but in environments where the nodes are active and perform tasks, they may offer a good way of monitoring whether a node is compromised. The authors of [33] use task success scores of UAVs to keep track of the number of tasks given, which were completed within the specified time allocation for each UAV.

Memory integrity, employed in [26], creates a hash of the memory of the node. The memory can be re-hashed and compared to the original hash to ensure that the OS/firmware has not been tampered with. In [26], the hash is immutably stored within the node, and a self-scrutiny method is used for the node to check itself. This method could also be employed in other ways, such as storing the original hashes of the nodes in a cluster head or blockchain, which could be compared to the memory hashes of any node at any time by other nodes in the network.

Other. Here, some more obscure measures seen in the models are discussed, as well as some methods that are not directly used for measuring trust but can be used to increase the security of the network and authenticate devices.

The authors of [11] define three perceptions used to calculate trust based on a probabilistic graph model (this is discussed further in Section 6). These perceptions are as follows:

- Ability—sensing and reasoning capabilities and influence on others.
- Benevolence—a measure of the trustor's belief about how likely the trustee is to assist the trustor.
- Integrity—the perceived characteristics of predictability, consistency, honesty and reliability of the CPS.

The authors of [34] use a somewhat similar approach, using three attributes modeled by generalized stochastic petri nets (another graphical probabilistic model type). The three attributes modeled here are:

- Reliability—the probability that the system will not fail.
- Availability—the probability that the system services are ready at a given time.
- Security—intrusion probability.

The authors of [38] define a set of good and bad manners with different impact values on the trust score of the node. These good and bad manners are mostly related to the parameters discussed earlier, such as communications, data quality, resource usage and behavior.

Techniques such as blockchains have been used in some models to immutably store publicly accessible data, such as trust values [9,33]. Authentication methods such as RF-DNA fingerprinting [9] and signature-based intrusion detection systems (IDS) [12] have also been seen to help to authenticate devices on static networks and prevent counterfeiting or sybil-type attacks. These techniques do not affect the trust calculation in the models, but they can be incorporated to add an extra layer to the security of the network.

6. Trust Model and Calculation Techniques

The inner workings of trust models are analyzed in this section. Methods for data preprocessing, trust calculation/classification, trust ageing and threshold calculations are all covered in this section.

6.1. Data Pre-Processing

Many models that follow the basic structure of Figure 1 calculate trust scores for various parameters in phase two of the diagram. These values are calculated directly from the trust parameters and can be calculated in a number of ways, depending on the data type. The values can be comprised of things including the direct measurement of parameters, such as bandwidth, message frequency, resource usage, etc., or something calculated through an equation, such as ratios of good to bad interactions, the sum of previous good/bad interactions, the error between the data given and the average data or data expected values, or a QoD score, to name just a few. This stage simply involves putting a numeric value on the parameters measured. We should note that these values are sometimes normalized in order to prevent having to deal with large numbers in trust calculations. From the research conducted, it appears that the way in which this pre-processing is carried out is not hugely important, as long as the numeric value is representative and correlates in some form (linearly, inversely, exponentially, etc.) with the way in which the trust is affected by the parameter.

Data may not always be transposed to a numeric value for further use. In some cases, the discretization of data may be carried out using fuzzy levels to split the data into a number of predefined levels, e.g., untrustworthy, neutral and trustworthy. Fuzzy trust levels are used in [18,27,28] to discretize trust values.

6.2. Trust Aggregation

With models that have pre-processed the data and calculated numeric values for trust for a number of parameters, often the models aggregate these values to create one final combined trust value. This is often achieved through either a weighted summation [3,9,10,16,22,34] (Equation (1)) or a weighted average [25,33] (Equation (2)):

$$\sum_{x=0}^{n} w_x X \tag{1}$$

$$\frac{\sum_{x=0}^{n} w_x X}{n} \tag{2}$$

where n is the number of parameters and w_x is the weight assigned to the parameter X. Usually, in Equation (1), the sum of the weights is set as equal to one.

Models often use static trust values, which can be set by the user upon setup. In some models, the weights used can be dynamically changed. The authors of [22] use an adaptive weighting system that is based on the number of successful interactions of the cluster and the feedback provided by the base station to dynamically allocate the weights. The authors of [16] provide a system where individual nodes can change the weight of the parameters based on their subjective opinions in attempts to provide a more versatile, flexible model that is not restricted to a single network.

6.3. Majority Voting System

An approach used in [21] is the majority voting system. In this network, beacon nodes (static nodes which provide location data) broadcast their communications for all the other beacon nodes in range to hear. The nodes then submit their vote on whether they think that a particular node is trustworthy or not. This system is simple but has drawbacks in that it assumes that the majority of nodes are uncompromised in the network, and it may not be able to resist ballot-stuffing or collusion-type attacks.

6.4. Probabilistic Models

Trust is not a hard measured metric; it can be quite subjective. Therefore, it makes sense that probabilistic approaches work well when classifying trust. Bayesian probability uses the frequency of some event (a trust measurement parameter) to determine the probability of some random variable (trust). A Bayesian formulation for determining a trust or reputation value is used in [12,20,25,27]. A probability distribution of the variable being calculated is needed for Bayesian methods. There are many types of distributions (beta, Gaussian, binomial, poison) that can be used depending on the circumstances of the model. Beta distribution is used in [20,25] and binomial distribution is used in [12].

Dempster–Shafer belief theory [15,20,24] is a similar probabilistic approach. This method is a framework for reasoning with uncertainty. DS theory can be used for calculating the recommendation trust of nodes from their neighbors. In [24], using DS theory, a belief function is used to represent the data presented by the recommenders, and the functions are combined using Dempster's combination rule. DS theory is also used to update recommendation trust in [20]. DS theory is used in [15] to combine all pieces of trust evidence obtained from the machine learning models and to define the upper and lower limits of trustworthiness for the data.

Markov chains constitute a system that changes from one state to another, according to a set of probabilistic rules. The probability of transitioning to any state in a Markov chain is only dependent on the current state and the time elapsed, i.e., it never depends on previous states or events [52]. In [38], trust is modeled as a sequence of states, with the lowest state being state 0 and the highest being state I. The events that will change the state are defined, and each event has an impact score. An impact of X will increase or decrease the state by X. The model uses this to determine the trust values of nodes based on their behavior and, if they drop below state 0, they are blacklisted. The Markov chain aspect in this case means that the current state of the node is never affected by any of its past states or actions.

6.5. Graph Models

Graph models use graph theory to model the network. Vertices in the graphs represent the nodes and edges represent the connections. The authors of [11,17] use probabilistic graph models which have a prediction probability for each node, defined as the probability that node k will detect the true state of the world. There is also a p-reliance and Q-reliance probability for each edge in the graph, which represent the positive and negative effects of the information exchanged between nodes, respectively. These probabilities are updated by obtaining the expected values and variance of certain parameters within the network.

6.6. Machine Learning Classification

Machine learning techniques can be used to classify nodes' trust levels based on the measured trust parameters. Any type of machine learning classification technique can be used to classify trust. Choosing the right one for the application depends on the typical factors, such as access to training data, the type of data, speed, computational overhead, accuracy, etc. Machine learning models offer very good classification results; however, they come with the requirement for significant computing resources, making them unsuitable for many network types. Networks with central nodes, such as base stations, with access to the required resources, are ideal candidates for implementing a machine learning model.

An interesting approach is taken in [23], where an unsupervised learning method, k-means clustering, is used to label a training data set. The reason for choosing this algorithm is its light computational overhead and speed. Then, a support vector machine is trained based on this labeled data set. The SVM is well suited to the sparse data set and is light computationally when compared to other supervised learning models. This process of training a model is carried out by a master cluster head node and can be updated with new data from the previous time window periodically. The trained model is then sent to cluster members, which can then use it to classify the nodes they communicate with as trustworthy or not, based on the trust parameters that they measure.

Random forest regression is used in [49] for classifying objective and device trust. The training data set is generated through simulations of the network and the labeling of the data is based on the average values of the simulated features.

Five types of trust evidence are split into three fuzzy levels, which are then classified using the C4.5 decision tree algorithm in [28].

A deep neural network is used in [15] to classify temperature sensor data based on its temporal correlation. The coefficients of the sensor data discrete cosine transformation are used as inputs to the networks. Two networks are used, one with a day as the time window for the sensor data, and one which further divides the day into smaller time windows. These two predictions are then combined using DS theory. The data set for training this model is generated through a six-month trial deployment of nine temperature sensors, and attack data is also added through six simulated sensors.

A genetic algorithm is used in [27] to optimize the trust model for detecting dishonest recommendations. The artificial bee colony (ABC) model is used and applied for detecting and isolating dishonest nodes.

6.7. Trust Ageing

Trust models sometimes use historical values of trust to build the reputation of a node or to factor it into the trust calculations. Of course, old trust values should not have as much bearing as more recent values; thus, some models take this into account by using methods such as weighting the old trust values less as they age [20], or using a decay function for the trust values over time [16,19]. This method allows trust to decay over time if no interactions are being created between the respective nodes.

The most commonly used form of managing old trust values is the sliding time window method [12,23,25,28,29]. This method uses a time window which includes the most recent trust values calculated (the number of trust values included can vary; here, we denote it as N). When a new trust value is calculated, it is added as the most recent value, and the time window "slides" over to include this value. Then, the N+1 most recent value is dropped from the window and is no longer used in any further calculations.

An interesting approach to trust ageing is taken in [16]. Along with the use of a decay function, here a trust maturity method is used. This method is based on the number of interactions between the given nodes and determines the point at which only the current direct assessments are necessary for calculating the trust between these two nodes. This means that there is no further need to gather the recommendation trust or rely on previous trust values for the rest of the session. This can reduce overhead in the trust calculation, increasing network efficiency.

6.8. Trust Thresholds

A trust threshold is the limit of the trust value according to which the node is considered trustworthy or not. One attribute that appears to be lacking in many models is a dynamic threshold. Most models employ a static threshold. Some offer a user-defined threshold [19,20,32]. Adding a dynamic threshold to a model creates a far more flexible model that can be used in more than one specific network case. For example, if two identical smart city network topologies are being used in two different cites, a trust model with a fixed threshold may not give the same results on both, due to differences in the network traffic, recorded data, etc. A dynamic threshold for trust could solve this issue. The authors of [3] use a dynamic threshold based on the average value of trust in the network.

7. Model Design Recommendations

This section discusses the design of new trust models and makes recommendations regarding what a trust model should include. Commonly, the most intensely studied aspect of the trust model is the trust calculation itself. This calculation should effectively establish a value of trust and/or classify data successfully as untrustworthy or not. This goal can be achieved with equally high accuracy in a number of ways. The parameters measured to obtain data in order to calculate this trust are perhaps more important. Table 1 shows a comparison of some of the models analyzed in this work. Clearly, data, communications and resource usage are the most common parameters used. This is due to the fact that every network includes these parameters in some form. They are essential for the operation of the network, which makes them a big target for many of the attacks outlined in Section 4.

Table 1. Comparison of the attributes used in trust models.

Model	Network Type	Measured Parameters				Trust Ageing	Dynamic Threshold
		Communications	Data	Interaction History	Resource Consumption		
[3]	WSN	✓	✓	✓			✓
[9]	IoT/CPS	✓					
[10]	IoT/CPS			✓			
[12]	WSN/IoT	✓	✓		✓		
[14]	IIoT	✓				✓	
[15]	IoT		✓				
[17]	IIoT	✓	✓				
[19]	MCS		✓			✓	
[20]	WSN		✓		✓	✓	
[22]	WSN	✓					
[23]	UASN	✓	✓		✓	✓	
[24]	WSN	✓	✓				
[25]	WSN	✓	✓		✓	✓	
[27]	WSN	✓			✓		
[28]	Underwater WSN		✓		✓	✓	
[29]	WSN	✓	✓			✓	
[30]	VANET		✓				
[31]	VANET	✓	✓	✓			
[32]	VANET	✓	✓	✓			
Usage Rate		68.4%	73.7%	21%	31%	36.8%	5.2%

When designing a new trust model, the parameters should be selected with attacks in mind. The attacks that the implementation of the model aims to mitigate should be

analyzed to determine what parameters of the network they will affect. Parameters behave differently, in most cases, while under attack; thus, measuring these parameters should help in detecting the attacks. For example, in a routing-type attack, the communication will be affected, and a DoS attack will affect the resource consumption.

All models should dynamically calculate trust, meaning that trust should regularly be updated for the nodes within the network. Trust is not a static metric and continues to change over time and should, therefore, always be updated regularly in order to detect malicious nodes successfully.

If a model is being designed with the intent of being used in more than just a single unique network, flexible design aspects should be included in the model to allow it to adapt to different networks. Machine learning models can achieve this through training on each unique network's data, or it can also be achieved by using methods such as adaptive parameter weighting [16,22] and dynamic thresholds [3].

For the most secure results, more than just a trust model is needed on a network. Trust models alone cannot provide confidentiality or data integrity. Encryption can be used to provide confidentiality and authentication to a network, and the hashing of messages can provide data integrity. Other methods for authentication can be used, such as device fingerprinting, which is used alongside a trust model in [9]. These methods can be costly for low-powered sensor networks in terms of computation and resource usage. Lightweight key management schemes and edge cryptography techniques can be used in these scenarios [36,37]. The use of the above techniques will enhance the security of the network but may not always be necessary, depending on the level of security needed.

Regarding simulators, we must note that the majority of works performed their testing and obtained their results on network simulators. The simulators used included:

- MATLAB;
- NS-2;
- NetLogo;
- COOJA (run on Contiki OS).

8. Trust Model Design Case Study: Factory 4.0 vs. Agriculture 4.0

Factory 4.0 and Agriculture 4.0 are subsets of industry 4.0, defined by the use of technologies such as AI, big data, robotics, blockchain and IoT [53] to help to improve efficiency, cost, production rates and yields in both factories and agriculture. Both Factory and Agriculture (Agric) 4.0 commonly use many of the network environments discussed in Section 3, such as WSN, unmanned ariel systems, IoT and cyber-physical manufacturing. There are many similarities between the two fields but also many differences. Trust models can be used to help to secure the network types used in each of these fields. However, due to the differences in network topologies and environments, the trust model designs differ. In this section, some design considerations for trust models are outlined for both Factory and Agric 4.0, and an analysis of the similarities and differences between the two is offered.

8.1. Network Environments

In this section, we discuss the Factory 4.0 and Agric 4.0 environments, which are generic setups/networks that apply to many but not all Factory and Agric 4.0 environments, respectively.

Factory 4.0 networks, in general, be in a smaller area than Agric 4.0 and are mostly indoors. The sensors have access to the infrastructure of the factory they are placed in. This can potentially give the sensors access to power sources and wired communication connections (this can all be achieved together using PoE or a similar technology). The sensors are difficult to access for unauthorized personnel due to the access and physical controls on the factory floors. Given these factors, we assume that power and communications are very reliable, and the probability of the sensors being taken down by physical attacks is low.

Agric 4.0 networks, on the other hand, have a more challenging network environment. The area of an Agric 4.0 network may be the entire farm (which, in large farms, will be thousands of acres of land), and the sensors may also have to deal with harsher environ-

ments. Agric 4.0 networks can be referred to as SAGUIN (Space-Air-Ground-Undersurface Integrated Networks). Sensors may be placed in all these harsh environments to monitor things such as soil/water/air condition and livestock. Along with these challenges, sensors may also have trouble accessing power, with options such as wireless power transmission, distributed wireless power transmission and ambient energy harvesting being considered to provide power to the sensors.

Communications in different areas of the farm need to be compatible with one another. For example, communications in the milking parlor or greenhouse may use Zigbee, or Bluetooth, as opposed to the communications in soil quality monitors throughout the land, which may need to use technologies such as LoRa for longer-range communications. These two networks most likely have to be combined in a central system for AI data analyses [53].

Table 2 gives a comparison of the Factory and Agric 4.0 network environments.

Table 2. Comparison on the Factory and Agriculture 4.0 environments.

Attribute	Factory 4.0	Agriculture 4.0
Area	Generally, a smaller area physically than Agric 4.0, which can consist of larger physical areas but, due to access to more infrastructure, such as highspeed reliable comms., it is not affected as much by physical distance	Larger networks with little access to infrastructure. Sensors often exposed to harsh environments
Communications	Use of wired and wireless protocols, such as PoE, WIFI, WIFI direct, Zigbee, etc. Generally, have access to reliable communications	Mostly using wireless communications. Often, long-range communications used, such as LoRa, 5G or radio
Power	Access to power sources due to nodes' proximity to infrastructure	Little access to power. Uses low-power systems with wireless/distributed wireless charging or ambient energy harvesting (solar, hydro, geo)
Physical access to sensors	Usually, nodes are at infrastructure that would be indoors or fenced off, with only authorized access permitted	Much easier to gain access to some sensors, as many are in open, easy-to-access fields/rivers

8.2. Trust Model Topology

With different environments come different network topologies. The network topologies should be optimized to suit the needs of the sensors/equipment. By analyzing the typical Factory and Agric 4.0 structures shown in Table 2, two network topologies were designed to suit each case's needs.

The Factory 4.0 network, as stated, often has more access to infrastructure, such as power and communications. This eliminates concerns regarding these limiting factors in the network. Often, the sensor and network nodes are fixed (they do not enter and leave the network frequently). An ideal network type for this would be a single (non-clustered) static heterogenous sensor network, such as the ones used in many of the IoT and CPS systems reviewed [9–11].

Agric 4.0 consists of a wider variety of networks and sensor types. There can be both large and small area networks in the same farm, and there can be unmanned ariel vehicles (drones), autonomous machinery and livestock. Agric 4.0 poses different challenges than most studies reviewed in this paper, and may thus need a unique network and trust model structure. A structure that could work for Agric 4.0 topologies would be a clustered hierarchical network structure with distributed trust calculations that can be combined and monitored at the base station. Clusters can be thought of as their own unique network, and trust can be calculated in a different manner for each cluster, e.g., one cluster could be a cluster for the long-range monitoring of crops and soil, while another could be the internal monitoring in a greenhouse or milking parlor. Each cluster can monitor its own network, ejecting malicious nodes, and the trust data can be reported back to the base station for monitoring of the trust within the entire network. This topology would be similar to the

structure seen in [3], with the exception that each cluster potentially uses a slightly different trust model. Figure 2 shows an illustration of the two proposed network structures.

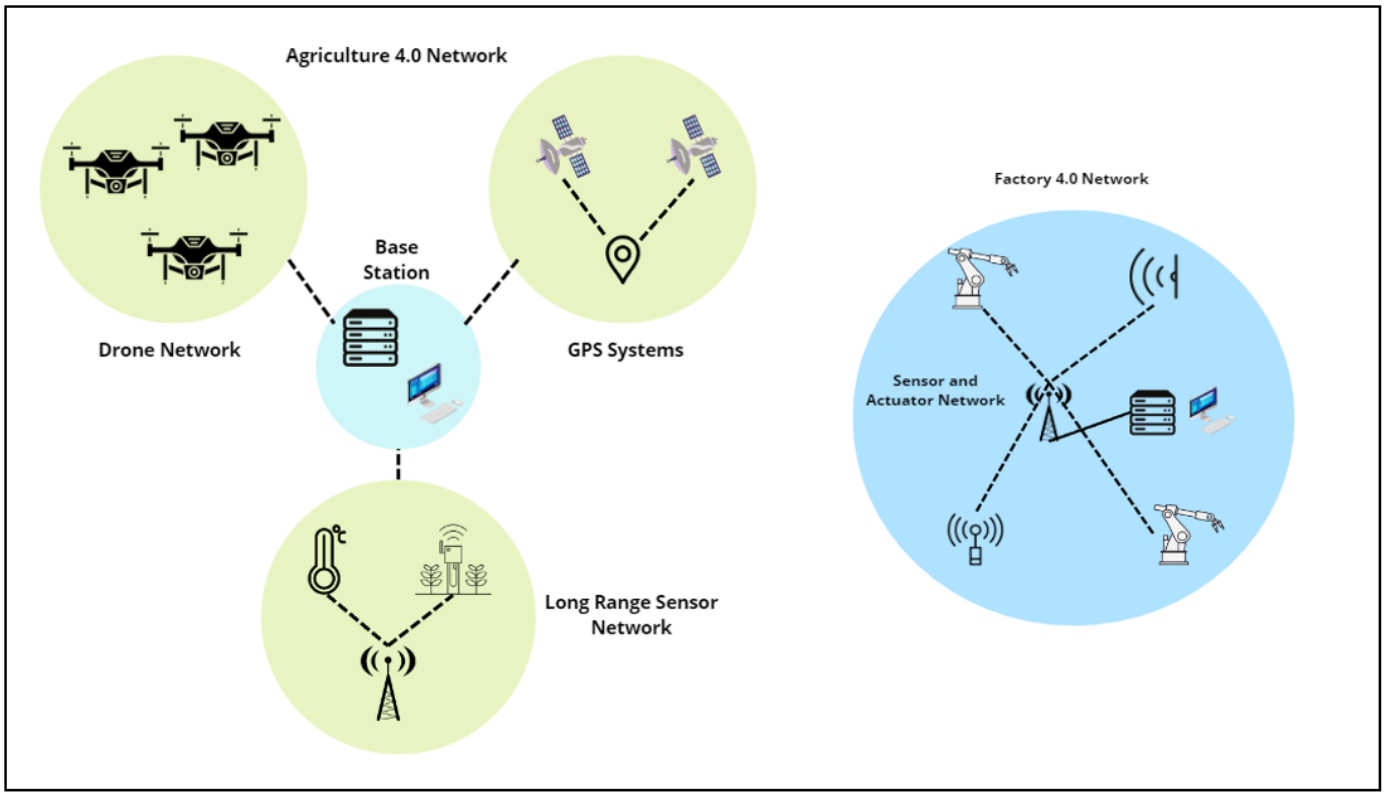

Figure 2. Comparison of Agriculture and Factory 4.0 proposed network structures.

8.3. Trust Parameters to Measure

As stated in Section 7, trust models should be designed with attacks in mind. The parameters selected to measure trust should be relevant to the attacks that are of concern to the network. Table 3 compares the two networks in the six attack areas discussed in Section 4 and identifies the key parameter/calculation used to combat each type.

Factory 4.0 is more like the typical network structure seen in the reviewed works, meaning the "typical" means of calculating and measuring trust, as seen in the reviewed works, can be applied. Agric 4.0 comes with more complexities and challenges. As we have seen, there can be multiple types of networks operating within an Agric 4.0 network. As previously seen in this work, different network types require different methods of calculating and measuring trust; thus, it makes sense that a complete Agric 4.0 network can be broken into multiple subnets. In each subnet, trust can be calculated and managed in a way that suits those subnets requirements. The subnets are still part of a larger network; thus, a proposed solution for this is to treat each subnet as a cluster with a cluster head that reports to a base station. With a structure such as this, there will be inter- and intra-cluster trust (similar to that seen in [29]). This network structure type could be classed as a hierarchical heterogeneous cluster network (every cluster is different) and could provide a good solution to the challenges posed by Agriculture 4.0.

Table 3. Comparison of attack types on Factory and Agriculture 4.0.

Attack Type	Factory 4.0	Agriculture 4.0	Parameter/Countermeasure
Access	High risk, as attackers will try to gain network access	High risk, as attackers will try to gain network access	Use authentication methods, e.g., fingerprinting. Monitor communications and data
Reputation	Dependent on trust model used	Dependent on trust model used—could be more common, as there may be a greater need for multi-hop networks over the large areas	If used, monitor recommendation trust values carefully and check for errors in reported values [24]
DoS	Communication DoS potentially more common, as devices will typically be less resource-constrained	Resource-draining in low-power nodes will cause them to die	Monitor comms and resource consumption
Payload	Dangerous in any network once attacker has access	Dangerous in any network once attacker has access	Monitor data/comms/resource consumption/memory integrity etc.
Routing	Depends on network topology	Potentially more vulnerable if multi-hop/larger networks are being used	Monitor comms - packet forwarding rates/successful interactions etc.
Physical	nodes may be difficult to gain physical access to	Nodes will often not be difficult to access	Try to minimize physical access to nodes, flag if a node has gone offline, disable any input ports on node device

9. Conclusions

Trust management and computation models provide a solution to help in securing sensor network environments against attacks that traditional security measures cannot defend against. This work reviewed the attacks in these networks that trust models can help to defend against, along with the parameters commonly measured in trust models and the methods of trust calculation from these measured parameters. Finally, the methods for designing a new trust model were outlined and demonstrated, with a case study comparing the trust model attributes in two separate network environments (Factory and Agriculture 4.0).

Future work in this field should consist of expanding on the current methods of trust calculation and on making trust models more dynamic in the way that they calculate trust and determine malicious entities in the network.

Author Contributions: Conceptualization, K.M., A.G. and T.N.; investigation, K.M.; writing—original draft, K.M.; writing—review and editing, A.G., D.O. and T.N.; visualization, A.G., D.O. and T.N.; supervision, T.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work has received funding from Science Foundation Ireland (SFI) under the Grant Number 16/RC/3918 (Ireland's European Structural and Investment Funds Programs and the European Regional Development Fund 2014–2020) and industry partner support provided by Analog Devices International, Limerick, Ireland.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Yousuf, M.; Kim, S. Coping with bad-mouthing in peer-to-peer file sharing networks. In Proceedings of the 2015 IEEE International Conference on Peer-to-Peer Computing (P2P), Boston, MA, USA, 21–25 September 2015; pp. 1–9. [CrossRef]
2. Dragoni, N. A survey on trust-based web service provision approaches. In Proceedings of the 3rd International Conference on Dependability (DEPEND), ACM, Venice, Italy, 18–25 July 2010; pp. 83–91.

3. Zhang, Z.; Zhu, H.; Luo, S.; Xin, Y.; Liu, X. Intrusion Detection Based on State Context and Hierarchical Trust in Wireless Sensor Networks. *IEEE Access* **2017**, *5*, 12088–12102. [CrossRef]
4. Ting, H.; Kang, X.; Li, T.; Wang, H.; Chu, C. On the Trust and Trust Modeling for the Future Fully-Connected Digital World: A Comprehensive Study. *IEEE Access* **2021**, *9*, 106743–106783. [CrossRef]
5. Tangade, S.; Manvi, S. A survey on attacks, security and trust management solutions in VANETs. In Proceedings of the 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, India, 4–6 July 2013; pp. 1–6. [CrossRef]
6. Guo, J.; Chen, I. A Classification of Trust Computation Models for Service-Oriented Internet of Things Systems. In Proceedings of the 2015 IEEE International Conference on Services Computing, New York City, NY, USA, 27 June 2015–2 July 2015; pp. 324–331. [CrossRef]
7. Farrokhi, B.; Nalbandian, S. A Survey on Fuzzy Trust Management in Cloud Computing. In Proceedings of the 2018 1st International Conference on Advanced Research in Engineering Sciences (ARES), Dubai, United Arab Emirates, 15 June 2018; pp. 1–7. [CrossRef]
8. Wang, D.; Muller, T.; Liu, Y.; Zhang, J. Towards Robust and Effective Trust Management for Security: A Survey. In Proceedings of the 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, Beijing, China, 24–26 September 2014; pp. 511–518. [CrossRef]
9. Kandah, F.; Cancelleri, J.; Reising, D.; Altarawneh, A.; Skjellum, A. A Hardware-Software Codesign Approach to Identity, Trust, and Resilience for IoT/CPS at Scale. In Proceedings of the 2019 International Conference 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), Atlanta, GA, USA, 14–17 July 2019; pp. 1125–1134. [CrossRef]
10. Mathas, C.; Vassilakis, C.; Kolokotronis, N. A Trust Management System for the IoT domain. In Proceedings of the 2020 IEEE World Congress on Services (SERVICES), Los Alamitos, CA, USA, 18–24 October 2020; pp. 183–188. [CrossRef]
11. Wang, Y. Trust Quantification for Networked Cyber-Physical Systems. *IEEE Internet Things J.* **2018**, *5*, 2055–2070. [CrossRef]
12. Meng, W.; Li, W.; Su, C.; Zhou, J.; Lu, R. Enhancing Trust Management for Wireless Intrusion Detection via Traffic Sampling in the Era of Big Data. *IEEE Access* **2018**, *6*, 7234–7243. [CrossRef]
13. Rani, R.; Kumar, S.; Dohare, U. Trust Evaluation for Light Weight Security in Sensor Enabled Internet of Things: Game Theory Oriented Approach. *IEEE Internet Things J.* **2019**, *6*, 8421–8432. [CrossRef]
14. Boudagdigue, C.; Benslimane, A.; Kobbane, A.; Liu, J. Trust Management in Industrial Internet of Things. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3667–3682. [CrossRef]
15. Karmakar, G.; Das, R.; Kamruzzaman, J. IoT Sensor Numerical Data Trust Model Using Temporal Correlation. *IEEE Internet Things J.* **2020**, *7*, 2573–2581. [CrossRef]
16. Adewuyi, A.; Cheng, H.; Shi, Q.; Cao, J.Á. MacDermott and X. Wang, "CTRUST: A Dynamic Trust Model for Collaborative Applications in the Internet of Things. *IEEE Internet Things J.* **2019**, *6*, 5432–5445. [CrossRef]
17. Wang, T.; Luo, H.; Jia, W.; Liu, A.; Xie, M. MTES: An Intelligent Trust Evaluation Scheme in Sensor-Cloud-Enabled Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2020**, *16*, 2054–2062. [CrossRef]
18. Li, W.; Cao, J.; Hu, K.; Xu, J.; Buyya, R. A Trust-Based Agent Learning Model for Service Composition in Mobile Cloud Computing Environments. *IEEE Access* **2019**, *7*, 34207–34226. [CrossRef]
19. Truong, N.; Lee, G.; Um, T.; Mackay, M. Trust Evaluation Mechanism for User Recruitment in Mobile Crowd-Sensing in the Internet of Things. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 2705–2719. [CrossRef]
20. Ganeriwal, S.; Srivastava, M. Reputation-based Framework for High Integrity Sensor Networks. In Proceedings of the 2nd ACM Workshop on Security of Ad-hoc and Sensor Networks, Washington, DC, USA, 25 October 2004.
21. Srinivasan, A.; Teitelbaum, J.; Wu, J. DRBTS: Distributed Reputation-based Beacon Trust System. In Proceedings of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing, DASC, Indianapolis, IN, USA, 29 September–1 October 2006.
22. Li, X.; Zhou, F.; Du, J. LDTS: A Lightweight and Dependable Trust System for Clustered Wireless Sensor Networks. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 924–935. [CrossRef]
23. Han, G.; He, Y.; Jiang, J.; Wang, N.; Guizani, M.; Ansere, J. A Synergetic Trust Model Based on SVM in Underwater Acoustic Sensor Networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 11239–11247. [CrossRef]
24. Reddy, V.B.; Venkataraman, S.; Negi, A. Communication and Data Trust for Wireless Sensor Networks Using D–S Theory. *IEEE Sens. J.* **2017**, *17*, 3921–3929. [CrossRef]
25. Wu, X.; Huang, J.; Ling, J.; Shu, L. BLTM: Beta and LQI Based Trust Model for Wireless Sensor Networks. *IEEE Access* **2019**, *7*, 43679–43690. [CrossRef]
26. Desai, S.; Nene, M. Multihop Trust Evaluation Using Memory Integrity in Wireless Sensor Networks. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4092–4100. [CrossRef]
27. Pang, B.; Teng, Z.; Sun, H.; Du, C.; Li, M.; Zhu, W. A Malicious Node Detection Strategy Based on Fuzzy Trust Model and the ABC Algorithm in Wireless Sensor Network. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 1613–1617. [CrossRef]
28. Jiang, J.; Zhu, X.; Han, G.; Guizani, M.; Shu, L. A Dynamic Trust Evaluation and Update Mechanism Based on C4.5 Decision Tree in Underwater Wireless Sensor Networks. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9031–9040. [CrossRef]

29. Khan, T. A Novel and Comprehensive Trust Estimation Clustering Based Approach for Large Scale Wireless Sensor Networks. *IEEE Access* **2019**, *7*, 58221–58240. [CrossRef]
30. Ahmad, F.; Kurugollu, F.; Kerrache, C.; Sezer, S.; Liu, L. NOTRINO: A NOvel Hybrid TRust Management Scheme for INternet-of-Vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 9244–9257. [CrossRef]
31. Ahmad, F.; Kurugollu, F.; Adnane, A.; Hussain, R.; Hussain, F. MARINE: Man-in-the-Middle Attack Resistant Trust Model in Connected Vehicles. *IEEE Internet Things J.* **2020**, *7*, 3310–3322. [CrossRef]
32. Xia, H.; Zhang, S.; Li, Y.; Pan, Z.; Peng, X.; Cheng, X. An Attack-Resistant Trust Inference Model for Securing Routing in Vehicular Ad Hoc Networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 7108–7120. [CrossRef]
33. Keshavarz, M.; Gharib, M.; Afghah, F.; Ashdown, J. UASTrustChain: A Decentralized Blockchain- Based Trust Monitoring Framework for Autonomous Unmanned Aerial Systems. *IEEE Access* **2020**, *8*, 226074–226088. [CrossRef]
34. Yu, Z.; Zhou, L.; Ma, Z.; El-Meligy, M. Trustworthiness Modeling and Analysis of Cyber-physical Manufacturing Systems. *IEEE Access* **2017**, *5*, 26076–26085. [CrossRef]
35. Jeong, S.; Na, W.; Kim, J.; Cho, S. Internet of Things for Smart Manufacturing System: Trust Issues in Resource Allocation. *IEEE Internet Things J.* **2018**, *5*, 4418–4427. [CrossRef]
36. Maimut, D.; Ouafi, K. Lightweight Cryptography for RFID Tags. *IEEE Secur. Priv.* **2012**, *10*, 76–79. [CrossRef]
37. Latif, M.; Ahmad, M.; Khan, M. A Review on Key Management and Lightweight Cryptography for IoT. In Proceedings of the 2020 Global Conference on Wireless and Optical Technologies (GCWOT), Malaga, Spain, 6–8 October 2020; pp. 1–7. [CrossRef]
38. Chang, B.; Kuo, S. Markov Chain Trust Model for Trust-Value Analysis and Key Management in Distributed Multicast MANETs. *IEEE Trans. Veh. Technol.* **2009**, *58*, 1846–1863. [CrossRef]
39. Tanabe, N.; Kohno, E.; Kakuda, Y. An Impersonation Attack Detection Method Using Bloom Filters and Dispersed Data Transmission for Wireless Sensor Networks. In Proceedings of the 2012 IEEE International Conference on Green Computing and Communications, Washington, DC, USA, 20–23 November 2012; pp. 767–770. [CrossRef]
40. Chen, Z.; Guo, S.; Zheng, K.; Li, H. Research on Man-in-the-Middle Denial of Service Attack in SIP VoIP. In Proceedings of the 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing, Wuhan, China, 25–26 April 2009; pp. 263–266. [CrossRef]
41. Kannhavong, B.; Nakayama, H.; Nemoto, Y.; Kato, N.; Jamalipour, A. A survey of routing attacks in mobile ad hoc networks. *IEEE Wirel. Commun.* **2007**, *14*, 85–91. [CrossRef]
42. Kavitha, T.; Sridharan, D. Security Vulnerabilities In Wireless Sensor Networks: A Survey. *J. Inf. Assur. Secur.* **2010**, *5*, 31–44.
43. Wang, C.; Feng, T.; Kim, J.; Wang, G.; Zhang, W. Catching Packet Droppers and Modifiers in Wireless Sensor Networks. *IEEE Trans. Parallel Distrib. Syst.* **2012**, *23*, 835–843. [CrossRef]
44. Palo Alto Networks. What Is a Denial of Service Attack (DoS)? 2022. Available online: https://www.paloaltonetworks.com/cyberpedia/what-is-a-denial-of-service-attack-dos (accessed on 8 January 2022).
45. Hu, Y.; Perrig, A.; Johnson, D. Wormhole attacks in wireless networks. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 370–380. [CrossRef]
46. Mathur, A.; Newe, T.; Rao, M. Defence against Black Hole and Selective Forwarding Attacks for Medical WSNs in the IoT. *Sensors* **2016**, *16*, 118. [CrossRef]
47. Kibirige, G.; Sanga, C. A Survey on Detection of Sinkhole Attack in Wireless Sensor Network. *arXiv* **2022**, arXiv:1505.01941. Available online: https://arxiv.org/abs/1505.01941 (accessed on 8 January 2022).
48. Grover, K.; Lim, A.; Yang, Q. Jamming and anti-jamming techniques in wireless networks: A survey. *Int. J. Ad Hoc Ubiquitous Comput.* **2014**, *17*, 197. [CrossRef]
49. Junejo, A.; Komninos, N.; Sathiyanarayanan, M.; Chowdhry, B. Trustee: A Trust Management System for Fog-enabled Cyber Physical Systems. *IEEE Trans. Emerg. Top. Comput.* **2019**, *9*, 2030–2041. [CrossRef]
50. Askham, N. *The Six Primary Dimensions for Data Quality Assessment*; DAMA UK Working Group: Bristol, UK, 2013; pp. 432–435.
51. Lear, D.; Droms, R. Manufacturer Usage Description Specification. 2018. Available online: https://tools.ietf.org/html/draft-ietf-opsawg-mud-25 (accessed on 26 January 2022).
52. Brilliant. Markov Chains | Brilliant Math & Science Wiki. 2022. Available online: https://brilliant.org/wiki/markov-chains/ (accessed on 30 January 2022).
53. Liu, Y.; Ma, X.; Shu, L.; Hancke, G.; Abu-Mahfouz, A. From Industry 4.0 to Agriculture 4.0: Current Status, Enabling Technologies, and Research Challenges. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4322–4334. [CrossRef]