1 INTRODUCTION

Security concerns will always remain a relevant topic of conversation. As technology innovations continues to excel at an astonishing rate, their design may satisfy only the general security requirements but that may not be enough. As devices, such as Internet of Things (IoT) devices, continue to be developed and deployed into networks, security concerns will continue to rise. Since there is such a variety of devices that can connect, utilizing a single security mechanism is simply not enough. In this paper, we attempt to survey research done on current and proposed mechanism for securing networks. While a single security mechanism is not ideal for every situation, much can be learned from the research that has been done to construct new security mechanisms and methods that can leverage new technologies such as Software Defined Networking (SDN) to create more secure and dynamic environments that can accommodate the diversity of devices and satisfy security requirements. Although these new technologies leverage many beneficial tools and features, they also pose new challenges as they are inherently susceptible to their own vulnerabilities.

2 IMPROVING WIRELESS PRIVACY WITH AN IDENTIFIER-FREE LINK-LAYER PROTOCOL

2.1 Overview

When talking about the link-layer, security is generally not one of the terms to come up in conversation. The reason for this is because the link-layer does not really address security concerns the way it could. As technology continues to move forward, it is crucial to ensure that everything that makes technology a success also moves forward, in parallel. This does not mean having to devise a completely new protocol, but it must at least be taken into consideration. The following paper was a fantastic approach to addressing a security issue that is present in link-layer communications. The authors propose an Identifier-Free Link-Layer Protocol, known as SlyFi, which aims to increase privacy by obscuring bits that are transmitted.

The main motivation behind the new protocol was to increase security across wireless technologies. The wireless ecosystem continues to grow at an alarming rate and it is becoming difficult to ensure that all these devices remain secure. Many security products are for endpoint devices, not dealing with the transmission of identifiers across foreign nodes. Since wireless device generally broadcast their messages and anyone listening can try to intercept these packages, there is a major concern for connection establishments and their authenticity. This lack of security across identifiers that are used to establish connections on wireless devices was the major motivation for devising a new link-layer protocol.

Although there are protocols, such as WPA, that attempt to address these concerns, they are only able to encrypt the contents of the message, not the identifiers in the header. With low-level identifiers still visible, an attacker listening to packets would be able to inventory, track, and profile users and devices without them even knowing. These attackers do not even have to be present to exploit this vulnerability making it a much bigger cause for concern. With these identifiers still visible, privacy concerns become an issue and, in a world, where people have many more than one device, the stakes continue to rise.

Their goal was to try and limit the linkability of communication packets. The authors first wanted to ensure that no information should be noted on the packet that would be able to tell someone who the sender or the intended receiver is. The other was avoid an attacker from profiling, fingerprinting, and tracking packets from the same sender. By doing so, they would able to completely obscure link-layer communication and increase privacy over wireless networks.

2.2 Security Requirements

To design a protocol that will be superior to those currently in rotation, the authors listed security requirements that were to be used as a guide. First on their list of security requirements is strong unlinkability. The motivation behind this was to ensure packets could not be linked

together for an attacker to track or profile them. In other words, this means is that only the parties communicating should be able to determine who the sender or receiver of a packet is. To accomplish this, the identifiers in the headers cannot be consistent and must be changed and randomized for every packet.

The next security requirement is in regard to authenticity. According to the authors, "to restrict the discovery of services to authorized clients and prevent spoofing and man-in-the-middle attacks, recipients should be able to verify a message's source. More formally, B should be able to verify that A was the author of c and that it was constructed recently (to prevent replay attacks)." [4]

Following this, we find confidentiality as the next requirement on their list. Essentially, the authors state that only the parties communicating should be able to see neither the payload of a packet nor the header and its fields. Last on the list we have message integrity. The authors want to ensure that anyone who receives a message should still be able to determine if it was tampered with in any way since it was sent from its original destination.

2.3 How it Works

By using their security requirements as a guide, the authors were able to devise a new protocol, known as SlyFi, that would satisfy all their requirements and even go beyond that. SlyFi is able to satisfy the security requirements by leveraging two mechanisms for concealing identifiers called Tryst and Shroud. SlyFI was meant to replace 802.11 by encrypting entire packets and then obscuring the header using pseudorandom identifiers. "A client wishing to join and send data to a SlyFi network sends a progression of messages similar to 802.11. Instead of sending these messages in the clear, they are encapsulated by the two identity-hiding mechanisms." [4]

Using SlyFi, if someone was looking to connect to a wireless access point, their device would transmit discovery messages, known as probes, that would be encapsulated using Tryst. These messages would be "encrypted such that: 1) only the client and the networks named in the probe can learn the probe's source, destination, and contents, and 2) messages encapsulated for a particular SlyFi AP sent at different times cannot be linked by their contents." [4] Once an access point receives one of these messages, it verifies the user that generated the message and makes sure they are authorized and also sends back an encrypted reply that will tell that users device that the access point is active. After this, if the user wants to establish a connection then an authentication request would be sent. This request would also be encapsulated using Tryst, but it would also contain information to help initialize the connection such as "keys for subsequent data transmission, which are used to bootstrap Shroud." [4] After a communication channel is established, Shroud will obscure both the payload and the header for everything sent. This will ensure that nobody will be able to see the destination or source addresses and be able to create a link to multiple packets as well as not being able to see the contents of the packets. Tryst and Shroud will essentially encrypt messages and their corresponding headers. Although Tryst and Shroud are rather similar in their functions, "the essential differences between them arise due to the different requirements of discovery, link establishment, and data transfer." [4]

According to the authors, "Identifiers are used in wireless protocols for two general functions: 1) as a handle by which to discover a service and establish a link to it, and 2) to address packets on a link and allow unintended recipients to ignore packets efficiently." [4] Both Tryst and Shroud are able to satisfy both of the requirements needed by unobscured identifiers.

When discovery or binding messages are used, Tryst will be the mechanism in charge of ensuring that only the devices communicating, and trusted network devices are able to decipher the header information as well as the contents of the message. Tryst is built upon symmetric key encryption and benefits from the infrequent communication and narrow interface features of symmetric key encryption. Since devices are

not usually sending these discovery and binding messages, they are considered infrequent. It is only when a device is searching for an access point that the messages are sent. Once established, they no longer need to be sent. Since the binding process is so quick, the narrow interface and short time of life allows for these messages to avoid being able to be linked together.

Once the connection has been established, it is time for Shroud to take over. The reason that a second mechanism is needed is due to the features that Tryst leverages in its mechanism which does not work for packages containing actual data. Shroud leverages public key encryption mechanisms to provide strong unlinkability for data transport packets. Shroud ensures that an address will never appear in two separate messages and allow them to be tracked by generating a pseudo-random identifier which is then used as an initialization vector for payload encryption.

2.4 Performance

The authors decided to use multiple protocols as benchmarks to compare against SlyFi. The protocols that were tested were wifi-open, wifi-open-driver, wifi-wpa, public key, symmetric key, and armknecht. Their first major test was to determine discovery and link setup performance by testing how long it took to establish a connection before a client was able to start sending data. Their results proved to be positive in comparison to public key and symmetric key, Tryst had "faster link setup times than wifi-wpa and scales as gracefully as wifi-open when varying each of these parameters." [4] To take things further, they managed to load multiple keys onto an access point which mimics an active access point. The reason for this is because an access point in a real-world environment would manage many keys which in turn would impact the link setup time. The results of this showed that Tryst was still faster that public key and symmetric key. As the number client keys on the access point went up, so did the time it took symmetric key to establish a link. Another portion of the test included the number of probes to unknown networks before a connection is established. The results to this were positive, showing no substantial increase in the time to setup as the number of probes increased unlike symmetric key and public key that both grew rather steadily as the number of probes went up.

The second major test was to determine how well Shroud performs in terms of data transport. Shroud was tested on both a software and a hardware platform to get a better comparison. The round-trip times of ICMP messages showed high latency for Shroud based on the software platform but on the hardware platform, it performed just as well as all other in the lineup. They believe that the reason Shroud based on the software platform was so high was because of an inefficiency in the Click runtime. Unfortunately, there was no further testing done to determine if this was true.

2.5 Observations

Overall, SlyFi is a well-documented and proposed identifier-free link-layer protocol that has great potential to increase privacy for wireless devices. It helps increase privacy by removing traceable identifiers from link-layer packets and using a unique proprietary mechanism to encrypt both the header and the payload. As the number of wireless devices in our society continues to grow, a protocol such as SlyFi becomes more and more important. Overall, I believe they did a fantastic job and the results speak for themselves.

3 SECURING INTERNET OF THINGS WITH SOFTWARE DEFINED NETWORKING

3.1 Overview

The Internet of Things (IoT) is becoming more relevant to networking technology considerations as time goes on. The reason for this is due to the number of devices that are being developed that allow users to connect to them from just about anywhere. On top of that, these devices are generally designed with minimal security considerations and can be vulnerable. Due to their inherent nature, networks must be able

to accommodate devices that not only generate big data and communication overhead but are also susceptible to many vulnerabilities.

The authors surveyed multiple Software Defined Networking architectures that were designed to extend security mechanisms into the IoT environment. They review each one individually and point out their benefits and their drawbacks. Based on the information they gathered, the authors then propose their own SDN architecture for the IoT environment that helps provide a more well-rounded solution with more of the benefits and less of the drawbacks.

"Since IoT has scalability and heterogeneity challenges, researchers and service providers have been exploring alternative solutions, such as SDN, to increase IoT's bandwidth and flexibility." [5] The reason IoT poses such a challenge for regular architectures is due to the diversity of devices and the fact that they are rather resource-constrained. IoT is also increasing at an alarming rate which in turn causes issues due to the amount of communication and the data that must be processed and stored. As more devices continue to connect, current architecture may not be able to handle the requirements that will be enlisted.

The main drive behind the paper was to leverage SDN to separate the control and data planes. This will free up many network devices because the SDN controller will oversee all decision making while switches will oversee just forwarding the data. Another issue was that in standard networks, routers are generally configured with individual mechanisms to handle security, link failure and many of the different forwarding mechanisms. When any of these need to be updated, management is done on an individual basis but by leveraging SDN, one would be able to establish a centrally managed system to combat this issue.

Essentially, SDN is being leveraged to "provide a network environment that is more dynamic and agile." [5] This will address the initial issues of heterogeneity and scalability of IoT devices. By separating the data and control planes and allowing the usage of less complex network devices as well as allowing for central management, SDN is able to provide many benefits and a feature rich toolset that can prove to be beneficial to the standard network. To avoid bottlenecking, SDN is able to provide logical routing and avoid this issue by maintaining a view of the entire network. . "SDN integration also simplifies information analysis and decision-making processes in IoT." [5] On another note, the authors note how SDN can be used in the IoT environment to help with troubleshooting and debugging due to the tools it naturally features.

3.2 Identity-Based Authentication Scheme for IoT Based on SDN

3.2.1 Overview: The first paper they surveyed proposed an "identity-based authentication scheme for IoT based on SDN." [5] The original authors aimed at leveraging the SDN controllers overview of the network to help determine any misbehavior on the network all while reducing the overhead generated by security-related information that is distributed across the network. The idea of implementing SDN into an IoT network environment is centralize control and be able to manage some of the complexities that will surface in this IoT environment. For this survey, the original authors decided to use IPv6 in order to have a suitable identity scheme. The SDN controller will be used to assign virtual IPv6-based identity to IoT devices. [8] For key establishment, Elliptic-curve cryptography (ECC) will be utilized as it rather efficient and is the most commonly used method for IoT.

3.2.2 How it Works: For this survey, the authors assumed "that the root controller public key is hardcoded in each thing when manufactured. "[8] The controller will create public keys for the devices that are using ECC while gateway will have its own set of public and private keys using ECC.

The first phase will require that the gateway send its ID and corresponding public key to the controller and will in turn receive a public key certificate signed by the controller's private key in order to authenticate itself to the controller

[8]. The second phase is where the devices will register with the gateway by sending an authentication request. This will include both that devices unique ID along with a random nonce. The nonce will then be used as the private key and be encrypted by the controller's public key. To prevent replay attacks, a second nonce will be sent. A preliminary identity check will be done by the gateway to verify whether the technology-specific address match one in the registration request. [8] If the initial verification is successful, the gateway then forwards the request, along with the corresponding certificate, to the controller where an IPv6 and a public key using ECC will be generated along with a private key that will be based on the random nonce that was received. The controller will then forward this to the gateway and from there, the gateway will "decrypt and store the hash identify and the corresponding public key." [8] The gateway then sends the "devices public key, the hashed address, and the gateway's public key encrypted by the devices private key." [8] The device will now verify registration with the controller and the authentication phase begins.

The third phase will be where the device will authenticate itself. It does this by sending a random none along with the hashed IPv6 address to the gateway. This will all be signed by the gateway's public key to ensure authentication. Once the gateway receives and authenticates, it sends back the original nonce sent from the device along with a second nonce and both will be encrypted separately using the device's public key. For the device to authenticate, it will return the second nonce, the one generated by the gateway during response, and encrypt It using the gateway's public key. The device is now authenticated and has permission to access the network. Since the device is authenticated, gateways can refer to the controller if the device were to move domains and need to connect to another gateway. This proposal was never deployed nor tested to try and see how well it operates in such an environment.

3.3 Host-Based Intrusion Detection and Mitigation Framework for IoT Based on SDN

3.3.1 Overview: The second paper surveyed is proposed as a "host-based intrusion detection and mitigation framework for IoT based on SDN", also known as IoT-IDM. [5] The main drive behind the survey was to devise a more secure framework that would allow for scalability while remaining affordable. Their proposal would allow for detection and mitigation at a network-level, rather than at the endpoints. One of their requirements during design was to devise a proposal that would detect attacks and then automatically trigger the appropriate countermeasures once the attack has been identified. While this is a fantastic idea, the authors had to ensure that efficiency was next on their list of design requirements to ensure overhead would not become an issue. Finally, they required that their proposal be scalable and be able to support new devices and their new technologies for both today and in the future.

3.3.2 How it Works: Their framework focuses on the use of two main technologies: Intrusion Detection System (IDS) and OpenFlow. The IDS can be either network-based or it can be host-based. In a network-based IDS, the traffic going across a network segment will be monitored and it will analyze traffic on the network and look for anything unusual or suspicious. On the other hand, a host-based IDS will only monitor a single-host in a network. This includes reviewing activities that the host is performing and remaining aware of its characteristics. The other technology being used is OpenFlow. This is being used "to enable the communication interface between the controller and underlying switches." [7]

The IoT-IDM architecture is broken down into five main modules: Device Manager, Sensor Element, Feature Extractor, Detection Unit, and Mitigation. [7] The Device Manager will maintain a database of all known devices and will include a risk assessment for each device. The assessment will note "potential security risks associated with them, the detection method of

known attacks, and if it is applicable, appropriate defense mechanisms." [7] The Sensor Element will be used to monitor a targeted device's activities while connected to the network by building a virtual inline sensor on top of the SDN controller. OpenFlow will be utilized to help create and distribute rules to try and redirect traffic from the device and network toward the sensor element. This will lower the amount of processing and load on the controller since only traffic for a targeted device will be passing through the sensor element and only the data collected from that targeted device is logged to be used later for auditing of incidents. The Feature Extractor used the data captured by the sensor element to determine the features of a specific device. The feature extractor is highly dynamic and allows for the extraction of specified data depending on the circumstances. Data captured and extracted by the feature extractor will be used later on by the detection unit. This system revolves around extensive knowledge of individual hosts on the network along with known possible attacks, but the problem lies with those attacks that may be unknown and how the system will handle them. The Detection Unit examines and identifies the data captured by the sensor elements and the information retrieved by the feature extractor to try and determine if there is any suspicious activity on the network. SDN is leveraged here to "enable machine learning algorithms to build predictive models for detecting malicious traffic" in the IoT-IDM architecture. [7] If the detection unit determines that suspicious activity has been detected on the network, it identifies the target and the source of the activity. From there, it will either send out an alert stating that suspicious activity was found but mitigation is not possible, or it will reveal any information regarding the activity to the mitigation module. The Mitigation Module is responsible for deploying countermeasures once a harmful attack has been identified. OpenFlow is leveraged to block or redirect the traffic flow and also allows a compromised host to be quarantined and provide limited access to the device.

3.3.3 Drawbacks: A main concern for this design is that since it is positioned on top of the SDN controller, "it is not technically feasible to use IoT-IDM for intrusion detection and mitigation process of all devices in a home network." [7] There would be too much overhead that can be added if too many devices were utilizing this architecture and would cause a rather negative experience if it were to be utilized on a larger scale. Also, since this is a host-based IDS, it means that the network is still susceptible to malicious traffic and protection is only guaranteed for a targeted host.

3.4 Black SDN for the Internet of Things

3.4.1 Overview: The third paper they surveyed proposed Black SDN for IoT, an SDN-based secure networking mechanism. The original authors chose to obscure all data that is transmitted across the network by encrypting both the payload and the header. Although this poses some challenges regarding routing the packets, the original authors utilized an SDN controller to control a broadcast routing protocol. This mechanism helps mitigate against analysis of the network's traffic but also against attacks that are focused on gathering data.

3.4.2 How it Works: Once a packet is completely encrypted, including both the payload and the header, it is considered a black packet. Black packets can only be routed by the SDN controller and the routes can be determined either statically or dynamically. In this design, the authors leverage the sleep and awake cycles of battery-constrained nodes. If the route is statically set, the route is predetermined and cannot be altered after transmission. If the route is dynamically set, a packet that comes across a node caught in sleep mode will be redirected to a node that is awake.

3.4.3 Benefits and Drawbacks: Although the idea of obscuring entire packets sounds appealing, the original authors did a poor job in designing the proper mechanism for obscuring the packets. Since the packets lack unique identifiers, only SDN controllers can route black packets and this

may be a cause for concern later down the road. Another issue is that since the controller is transmitting and routing against nodes depending on their sleep cycle, communication overhead is incurred. Overall, the concept was there but the original authors failed to present an ideal mechanism for obstruction of entire packets including header and payload.

3.5 Flow-Based Security for IoT Devices Using an SDN Gateway

3.5.1 Overview: The fourth paper they surveyed proposed "a flow-based security approach for IoT devices using an SDN gateway." [5] This is done by configuring the SDN controller as a gateway and then monitoring all traffic that flows across the controller. If any suspicious activity is found, a response for the corresponding flow will be triggered. The main drive behind this proposal was the "need for a flexible and dynamic method of IoT security." [2]

3.5.2 How it Works: The SDN controller acting as a gateway is made up of three main components: switch functionality, a statistics manager, and mitigation actions. The statistics manager is in charge of data collection and being able to characterize flows in order to help detect anomalous flows. When an anomaly has been detected, an action is selected from the mitigation action list and is enforced. "The three possible actions are; Block, Forward, or Apply QoS." [2] Block will prevent a device from communicating across the network by blacklisting it. Forward redirects the traffic to a quarantined section that is configured on the network until further review can be done to determine the appropriate actions to take with the device that triggered the mitigation action. Apply QoS is used when the controller cannot determine what the best choice is or if it is unable to block a single source. This will limit general service requirements such as bandwidth in order to limit the flow.

3.5.3 Benefits and Drawbacks: While this proposal may have lacked valuable information to validate its effectiveness, the original authors were able to propose an additional option to use SDN in order to increase security. The original tests that were done using this design were only using Transmission Control Protocol (TCP) and Internet Control Message Protocol (ICMP) DDoS attacks. Because of this, the design does not have sufficient testing done for validation of its effectiveness.

3.6 SDN-Based Architecture for IoT and Improvement of the Security

3.6.1 Overview: The fifth paper the authors surveyed proposes a secure SDN-based solution by utilizing multiple controllers for a single OpenFlow enabled switch. The main drive behind this design was that current mechanisms that are deployed on the Internet edge are not sufficient to handle the future of the Internet. "The borderless architecture of the IoT raises additional concerns over network access control and software verification." [3] This design utilizes multiple SDN controllers to be synced together and setup "in a security perimeter enabling a granular control over network access and continuous monitoring of network endpoints." [3]

3.6.2 How it Works: The original authors of this design set forth guidelines to ensure their design would be successful. The first issue they addressed was the single-point-of-failure issue and this was done by utilizing multiple controllers where there will always be a controller ready to take over in the event that a controller was to fail. OpenFlow enabled switches are then configured to more than one controller to satisfy the single-point-of-failure concern. Since there are multiple controllers, OpenFlow can either be in equal interaction mode or master/slave interaction mode. With equal interactions, read and write access on the switch is given to all the controllers which will require them to sync up in order for them not to overlap during read or writes. With master/slave interaction, one controller will be labeled as the master and all others will be labeled as slaves. This prevents controllers trying to read or write over each other.

For their design, they classify nodes into two categories: nodes can be considered an OpenFlow node if they have the resources available

and if they don't, they are considered a smart object. Regarding SDN domains, each one will have a dedicated SDN controller that will handle all traffic within its domain. The authors then propose a new type of controller, known as the root controller or border controller, to achieve large scale interconnection across multiple SDN domains. [3] This controller will be responsible for communication and connection across other SDN border controllers. Another item to note is that the authors chose not to distribute control functions and instead decided to distribute the routing functions and security rules of each border controller. [3]

During the initial setup, the OpenFlow switch will be authenticated by the SDN controller. Once authenticated, "the controller blocks switch ports directly connected to the users." [3] Then, the controller only attempts to authenticate the users by only allowing the users authenticate traffic to flow across the network. Each user is given a level of authorization so once the user passes authentication, the controller will look at what their level of authentication is and then push the corresponding flow entries to the access switch. [3] For IoT devices, each must be associated with an OpenFlow enable node that is connected to at least one controller within the domain.

3.6.3 Benefits and Drawbacks: While this design allows for each domain to remain independent in the event of a failure, its overall structure will cause communication delays. The design also revolves around the Grid of Security concept which essentially allows SDN domains to remain independent and enforce whatever security rules they choose but when a node wants to communicate with another node in separate domain, the flow must be directed to the border controllers, otherwise considered the security controllers, and then that controller is in charge of communication, only allowing recognized traffic and a safe connection. Unfortunately, the original authors did not do any performance testing which gives no information as to how much communication overhead is truly being generated with this design.

3.7 Proposed Architecture (Rol-Sec)

3.7.1 Overview: After surveying the proposals, the authors concluded that these mechanisms could be classified into three different categories: network-based, traffic-based, and crypto-based. The network-based mechanisms were designed to "deal with the architectural structure of the network elements." [5] Traffic-based mechanisms were mainly focused on flows and were there for detection and prevention of malicious activity on the network by analyzing flow on the network. Crypto-based solutions provided security by leveraging encryption as the focus of their proposal. By utilizing the advantages and drawbacks of each mechanism, as well as utilizing the classifications as a guide, the authors propose their own take on a mechanism that leverages SDN to secure IoT devices and environments.

The authors proposed Rol-Sec, a role-based security controller for the SDN-IoT environment. By assessing the advantages and disadvantages of each proposal they reviewed, the authors were able to devise a mechanism with minimal drawbacks. Rol-Sec consists of utilizing an SDN-IoT gateway (SDN-IoT GW) in each domain that will not only communicate within its own domain but will also communicate across multiple domains via other SDN-IoT GW's while being managed by several controllers.

For their architecture, the authors decided on using role separations of the controllers in order to try and distribute overhead generated by IoT data. The three controllers will be an intrusion controller, a key controller, and a crypto controller. Each one has an individual task and essentially distributes the overhead across multiple controllers.

The intrusion controller is responsible for availability and secure routing. Availability is accomplished by analyzing the flow and the corresponding rules for each flow, along with monitoring the general traffic. If some sort of intrusion were to be sent into the network, the intrusion controller would be able to detect it and then attempt to mitigate it. As for secure routing, the intrusion controller will attempt to find not only the

best possible route but also a route that is secure. This is done by communicating with the key controller and receiving whatever keys may be needed for the utilized algorithm.

The key controller will act as the key manager will both store and distribute keys. On top of acting as repository and a trusted third party, the key controller is also responsible for cryptographic operations that may be needed by either the intrusion controller of the crypto controller. This controller can communicate with both the intrusion controller and the crypto controller.

The crypto controller is responsible for ensuring "integrity, confidentiality, privacy, authentication, and identity management." [5] Different cryptographic algorithms are leveraged to "take into consideration the dynamic, heterogeneous, and scalable characters of the IoT environment via the SDN controllers." [5] Since the crypto controller require keys during its operation, it must be able to communicate with the key controller.

3.7.2 Benefits and Drawbacks: By designing a distributed architecture that is devised of multiple controllers with specified roles, the authors were able to mitigate the single-point-of-failure issue. For this design, the authors also stated that since the keys will be "stored and distributed over different communication links, they do not need memory storage on other controllers and they do not affect cryptographic operations or intrusion detection or prevention mechanisms." [5] Overall bandwidth is improved as well since the communication traffic will be distributed and not originating from a single point or communication link.

Since their design is accomplishing by role separation and the distribution of multiple controllers, communication channels across the devices will need to be used instead of single localized system that has the information in memory. Because of this communication requirements, their design incurs additional latency and communication overhead. To mitigate this, the authors suggest utilizing powerful devices that can process at high rates along with utilizing high speed link connections.

3.8 Observations

Although each proposal has its own advantages and disadvantages, there are still concerns for the future of SDN-IoT security. One of the concerns is that SDN is susceptible to the single-point-of-failure issue and it becomes an issue since SDN is centrally managed. This means that if a controller was to have an availability issue, every single system in the architecture will become unavailable as well. Another concern is that the limited table sizes of the switches will become an issue as the number of devices in the environment grows. This is because unknown flows are individually inserted as a new entry into the memory of that switch and as the number of devices added to the environment increases, efficiently trying to store the drop and forward rules becomes increasingly difficult. Further concern surrounds the possibility of causing a bottleneck in the system between the gateway and the controller as the number of IoT devices increases. Although there are challenges for the SDN-IoT environment, it is fair to say that SDN can be leveraged to aid security requirements in such an environment as long as consideration of its inherent vulnerabilities are noted, and the appropriate measures are taken to reduce their risk.

4 SECURITY ISSUES AND CHALLENGES IN WIRELESS SENSOR NETWORKS: A SURVEY

4.1 Overview

Sensor nodes are the basic building block of wireless sensor networks (WSN). The components of a sensor node vary based on the context and situation. The units of a sensor node generally include memory, processor, power, sensor, transceiver, position finding system, and a mobilizer. The role of the transceiver unit is critical to the sensor node as it is responsible for sending and receiving data and/or signals. The transceiver also accounts for more power consumption than the other units so as sensor nodes are being designed, the transceiver should be designed to reduce energy consumption when sending and receiving data. While the market

offers different sensor nodes, the nodes tend to have a small memory space, limited processing power, and a small flash ROM to compile processing data. A general reference architecture defines the core functionalities, standards, and the application program interface (API). APIs facilitate an application developer to interact with the system services. The WSN reference architecture includes Physical hardware/Sensor nodes, Operating System/Firmware, System and Storage Services, Programing Abstractions and APIs, and Applications. A WSN is comprised of multiple sensor nodes with individualized roles. The Sensor node senses the events that are collected by the aggregator-nodes. Then, the aggregator-nodes forward all events to the Base Station where it will forward all events to a remote server directly or via the Internet. The role of the Base Station node is vital in WSN as it is responsible for the collection of critical data, topology generation and malfunctions in the network. The aggregator node is responsible for hiding the base station node from inside and outside attackers, along with reducing delays and increasing network availability.

4.2 WSN Applications

Today, WSN's are used in industry, society, agriculture, wild life, and environmental protection arenas just to name a few areas. WSN are interconnected sensor devices that produce a huge amount of data, however they are also resource constrained devices. Sensor devices are used in buildings, college campuses, road traffic monitoring, vehicle tracking systems, atomic reactors, and other mission critical systems.

WSN's in the industry arena include such categories as gas sector, physical and environmental where sensors monitor temperature, humidity, moisture in the soil, wind, and pressure to improve efficiency. WSN facilitates society in areas such as traffic monitoring and even health care. With the help of WSN, doctors can diagnose a disease more precisely using a body area network to monitor body temperatures, blood pressure, sugar, and stress levels. The WSN could also lead to smart cities where everything is interconnected. Smart homes, buildings, and even bridges can leverage WSN to reduce energy consumption and monitor reliability of structural materials used in building and bridges. In the wild life arena, WSN can be leveraged to track animal's behavior such as how the interact with others and their environment as well as their habits. [1]

In agriculture, WSN's can monitor many important variables needed for successful crops such as different soil types, temperature, winds, water quality, humidity, and sunlight intensity. All of these variables are important and can affect the agriculture systems if they are not monitored accordingly. This scientific approach to agriculture leads to smart irrigation systems that reduce overall excess waste of water. In a disaster situation, sensor nodes can be deployed to gather data and have a better overview of the situation and the appropriate mitigation steps that should be taken. Pre-installed sensor networks can be leveraged to notify the proper authorities in the event that something such as a forest fire or a flood were to occur. WSNs are also used for environmental protection efforts in by monitoring things such as atomic reactors, volcanos, and pollution. [1]

4.3 Security Requirements

WSN's revolve around three basic security requirements: confidentiality, integrity and authentication. When WSNs are designed around the basic security requirements, network stability as well as operations will become optimal. While cryptography can help to protect the system, a secure network should consider authorization and authentication. Authorization will allow data access only to legitimate users and authentication will ensure that the data is from the right source. A strong authentication reduces attacks related to authorization. For this survey, the authors categorized WSN security requirements into four levels: data level security, node level security, code or program level security, and network level security. [6]

4.3.1 Data Level Security: In WSN's, data that is in transit can be captured by adversaries who can then alter the captured data and attempt to resend it. Data level security defends against corruption and modification but also guarantees protection from unauthorized access. Data level security in WSN relies on authentication, confidentiality, integrity and freshness. Authentication ensures that the data is coming from the correct source. This can be accomplished by incorporating a username and password, using tokens, digital certificates, or smart cards. Confidentiality refers to assuring that only authorized nodes can access data and unauthorized nodes cannot. This protects the content from being accessed by someone in which it was not directed to. while assuring the privacy of the data. Integrity deals with the authenticity of the information. Is the information accurate, compete, and trustworthy? Freshness ensures the that old messages cannot be replayed, and that the information is the most recent. Whenever packets are received out-of-order, the freshness is known as weak freshness. Packets received in-order is called strong freshness. At the data level, we want to strive for strong freshness of all data.

4.3.2 Node Level Security: Due to the hostile nature of the environment some situations will require more security. Some instances include military operations, fires in wild-life, cold temperature zones and atomic reactors. Proper node level security protocols will avert adversaries access to a sensor node and to obtaining the cryptographic keys and the underlined designed secure protocol. To ensure node level security, consideration must be given to availability, authorization, non-repudiation, and secure localization. At node level security, availability refers to the information being accessible to the parties who need the information and are also authorized to receive it. Critical and logical phenomenon's such as high temperatures, communication issues, node failures, floods, and storms will affect availability but WSN's should ensure that nodes are available at any given time. If a network is not available, it is deemed useless regardless of how strong it may be. WSNs should ensure that all nodes and gateways must be available all the times. Authorization, or a right to access, is controlled by the network administrator. In WSN's, the network administrator assigns different rules and polices for users based on their specific needs. Some users may require access to modify information while others may only need access to read the information without having to modify it. At the same time, some nodes have access to read data, some nodes can send data to the base station, and some nodes cannot do either. Authorization is a critical aspect of WSN's and it is generally controlled by the network administrator. Non-repudiation in WSN's prevent a node from later denying that is was involved in communication. Non-repudiation also provides the digital evidence needed to prove the origin and integrity of the data. In WSN's, secure localization refers to being able to locate a sensor node in the network. The ability to do so automatically and accurately is crucial when designing a sensor to locate a fault and analyze errors in a network.

4.3.3 Network Level Security: Unauthorized data or system access is controlled by a network administrator who manages the overall network level security. The network level security is comprised of deploying and enforcing policies and rules that will ensure the safety of the network hardware and data, scalability, reliability and integrity for the network. To be effective, network level security in a WSN must consider self-organization, time synchronization, scalability surrounding security, energy consumption, performance. In a WSN, self-organization can be leveraged to try and mitigate certain vulnerabilities a sensor network is inherently susceptible to such as eavesdropping, man in the middle attacks, and denial of service attacks.

Regarding time synchronization, it is a way of synchronizing sensor nodes with the centralized sensor node to avoid replay packets and to ensure that sensor nodes guarantee real time data and data freshness. Coordination of time between sensor nodes is also vital in sensing events such as temperatures, humidity, and movement.

Scalability is important in a WSN of the number of sensor nodes is generally large and it is important to ensure that the system can continue to operate efficiently as the network grows. WSN's must be built with the capabilities to seamlessly incorporate new nodes, when and if needed, without any adverse effects on the network. Another important aspect of network level security is surrounding security, which consists of proper surveillance mechanisms that help monitor malicious activity such as surveillance cameras and sensors. Surrounding security may be more practical in some fields than others, such as being utilized for wild life monitoring and industrial applications, but not so much in military operations involving monitoring the enemy. WSN's inherently suffer from power consumption and their resource constraints. Lack of, or even insufficient, power can result in the node becoming unavailable or even drop messages as there is not enough power to forward. Because of this, sensor nodes in the network must be designed to conserve energy whenever possible. Often time a tough choice must be made between performance and energy consumption in the network. For example, multifaceted cryptography applications involving high level security will be a detriment to the senor battery life, while low processing operations such as measuring temperature and humidity will conserve more energy.

4.4 Operating System and Tools for WSNs

In any computing system, the operating system (OS) is in charge of governing all operations.). A robust OS is necessary in WSN's to deal with standard issues that may arise but more importantly to address security holes. Some key points to note is that the OS may provide system level security include such as granting access or giving permission, memory protection, and enabling privilege mode. Due to the limited resource constraints of WSN's, those in charge of designing the OS will be faced with challenges. To overcome some of these challenges, re-programing may be used to while also being able to provide more flexibility. By Implementing re-programing, we can see how adaptable and flexible the system is to changes that occur in the network. A system that has static behavior can be changed to dynamic behavior through reprograming. The roles of wireless network tools are crucial to the operating system to manage, configure, and update the sensor nodes. These tools also play a vital role in programing, examining, analyzing media, and troubleshooting the entire network.

4.5 Attacks

As discussed earlier, WSN's have numerous applications but are inherently susceptible to a long list of possible attacks. The attacks on WSN's are broken down into several groups: physical attacks, communication attacks, code attacks, base station attacks, and routing protocol attacks. Within those groups, some attacks include data alteration, node stealing and tampering, traffic tracing, and tampering. Communication between nodes can also be monitored by adversaries who may attempt to steal data.

The primary objective of physical attacks in WSN's is to remove or physically damage the sensor. Once the sensor is removed and is in the possession of the attacker, he can analyze the internal code and extract information such as the cryptographic keys and then later implant the modified node back into the network which will lead to future attacks where the attacker identity is concealed. Some examples of physical attacks in WSN's are new node injection attacks, Sybil attacks, reverse engineering attacks. To give an example, a Sybil attack is when an attacker attempts to make the node have various identities which will it difficult to tell the difference between a legitimate node and a phony node.

Communication attacks in WSN's are considered logical attacks where an attacker can corrupt or alter the sensor node remotely. Some instances of communication attacks in WSN's are jamming, DoS, and collision attacks. A jamming attack is when an attacker attempts to impede, or jam, the communication signal. This attack is usually against the physical layer in a WSN architecture.

Code attacks in WSN's refer to vulnerabilities at the application level such as vulnerabilities within software programing and coding. Some cases of code attacks in WSN's are overflow attacks and changing the existing node remotely by re-programming. In an overflow attack, the attacker overflows the buffer in order to make the node become unavailable and unable to make new communications.

Base station attacks in WSN's attempt to gather data sent to the base station from all the surrounding nodes. Some common base station attacks in WSN's are source location attacks, destination location attacks, traffic analysis attacks, traffic tracing attacks, and content analysis attacks. If the base station were to be compromised, all the sensor nodes and associated deployment cost will also be compromised.

Routing protocol attacks in WSN's are caused by weak protocols which could give an adversary access to privileged mode. Some examples of routing protocol attacks in WSN's are black hole attacks and HELLO flood attacks.

4.6 Defenses/ Countermeasures

To prevent the attacks on WSN's, multiple approaches must be utilized to protect WSN's from attackers. These approaches provide data and destination privacy while preventing physical attacks, communication attacks, code attacks, base station attacks, and routing protocol attacks. To guarantee business privacy and integrity cryptography techniques must be used. However, it is difficult to apply traditional passwords techniques such as two-factor authentication in WSNs. It is recommended that when implementing a cryptographic approach in defense of confidentiality, integrity, availability, to use a light weight protocol. Sensor nodes are surrounded by wireless signals which creates a huge impediment to implement an adequate security system. Noise can affect the signal strength and it is impossible to prevent signal manipulation. Therefore, it is advised to deploy WSNs in a non-susceptible noise area. Regarding securing a wireless link, it is almost inconceivable to avoid signal propagation. Availability is one of the key pillars of WSN's and is there to ensure that data should be available when needed. Some approaches in defense for availability are implementing redundancy for substitute and failure, adequate power routing protocols, crisis energy backup, and guarding against detrimental actions such as DoS attacks. Application level defense in WSN's requires compact code for resource constraints devices like sensor node and ensuring that there are no vulnerabilities within the actual programming of the application.

4.7 Challenges

There are several challenges when it comes to WSN's. WSN's must be designed accordingly to address performance, cloud computing, reliability, SDN integration, scalability, and virtualization. In addressing these challenges, focus must be given to having strong routing protocols that handle mobility nodes and consider base station privacy. Key management and distribution are also an important challenge for WSN's and will continue to be an area of research.

4.8 Observations

There are several challenges when it comes to WSN's. WSN's must be designed accordingly to address performance, cloud computing, reliability, SDN integration, scalability, and virtualization. In addressing these challenges, focus must be given to having strong routing protocols that handle mobility nodes and consider base station privacy. Key management and distribution are also an important challenge for WSN's and will continue to be an area of research. Overall, I feel the researcher did an outstanding job of providing an in-depth analysis in the area of wireless senor networks. The future for WSN is promising. However, better techniques are needed for security, privacy, power, computational-capability, and scalability. Software defined networking integration in WSN will transform the architecture of WSN. Benefits and challenges of cloud services and virtualization technology also need attention. After surveying this paper we've gained a better and

understanding and comprehensive knowledge in WSN.

5 A SURVEY ON SECURITY THREATS AND AUTHENTICATION APPROACHES IN WIRELESS SENSOR NETWORKS

5.1 Overview

Sensor networks distributed wireless technology are used in numerous applications. Sensors measure a specific data point like temperature or magnetic field. Due to having low processing overhead, their cost is low as well. A WSN also requires lots of sensors, a data center, and a base station.

Current work on sensors focus on designing them to withstand harsh weather conditions, decrease overall energy consumption, lowering the cost to produce individual sensors while maintaining high capacity, and improving overall security and speed of data flow.

Once sensor nodes have been placed, they are unobservable and manual storage is impossible. This can leave important data vulnerable so it important to implement information security precautions. Wireless sensor networks often rely on broadcast messages. Sensor nodes must also be able to authenticate other nodes along with their messages while ensuring that routing is secure.

5.2 WSN Architecture

WSN's rely on multi-hop transmission to transmit sensor data to a data center safely and quickly. The final node to communicate with the base station is called the sink node. The order of transmission is sensor node, neighboring nodes, sink node, base station, data center.

5.3 Types of Attacks on WSNs

Because WSN's operate over wireless, they are exposed to more attacks such as those that can alter routing amongst nodes, create fake messages, and increase delays in the network; these methods are sometimes called spoofing.

Another form of attack is selective forwarding in which a node is controlled by an attacker as if it were a black hole. This node will then destroy messages or has them stop forwarding to cause jams and data collision within the network.

Sinkhole attacks are one type of selective forwarding. In this instance, the bad node is like a sinkhole. This sink node pulls all data toward itself and convinces other nodes to receive its tampered packets. A sinkhole attack provides the setting for a wormhole attack which is when two nodes are convinced they are neighbors. These two nodes will be responsible for the attack where one node is on the network neighboring the base station, and one is a neighbor of the target node which ultimately creates shortcuts within the network.

A Sybil attack is when one node, the Sybil node, takes on the identity of other nodes. This decreases the safety of fault tolerant systems, changes location information, and can also begin selective forwarding attacks.

In a HELLO flood attack, the attacker node deceives other nodes into believing they are neighbors. Upon the receipt of nefarious HELLO packets, the network experiences confusion. During an acknowledgement spoofed attack, an attacker preys on dead nodes by telling them they are still alive and receiving their data. This can be a gateway to selective forwarding as well.

5.4 Authentication Protocol

WSN's require both cross-node as well as node-to-center authentication to ensure that nodes recognize one another and trade data in a secure manner. Elliptic Curve Cryptography (ECC) is generally used for security protocols in WSN's to safeguard against identity impersonation attacks. The Sensory Network Encryption Protocol (SNEP) adds encrypted text as a message authentication code (MAC) before a chaining encryption. This prevents attackers from translating any plain text. The base station uses the TESLA protocol to derive the MAC. When a node is receiving a packet, it must ensure that the MAC key is intended for the base station. First, the message is placed in intermediate memory. When the receiving note declares the key, the base station then broadcasts all the MAC keys. If it is

a true key, the package is moved to the buffer cache. Security Protocol for Sensor Networks (SPINS) communications secure wireless sensor networks using both TESLA and SNEP. The Localized Encryption and Authentication Protocol (LEAP) provides confidentiality and authentication by equipping each node with four types of keys: the individual key, pairwise key, cluster key, and group key. One disadvantage of LEAP is the assumption that the sink node will not be compromised. [6]

5.5 Observations

Energy efficiency, security, and routing are some of the challenges a WSN will face. Of those, security seems to pose the biggest challenge to WSN's as the innovation is just not there. Due to the amount of overheard that can be incurred with new security mechanisms, finding the balance proves to be an issue that goes unresolved. The level of security needed will be based on the application, with some applications requiring greater security than others. Security is a process that should begin at system origination and remain updated as the attacks continue to advance and new attacks are uncovered. Current comparisons and work in the field show that the authentication methods are commonly used, yet further study of key infrastructure and complexity are necessary in future research.

6 CONCLUSION

Moving forward, network security is expected to remain a top priority. With the use of WSNs, cloud implants, reigning in IoT, and increasing automated attacks, the role of the network security administrator will likely change based on the types of threats in the future, as well as advances in the technologies and processes organization and government use to combat those threats. On the other end, SDN is beginning to prove itself as a viable solution to helping handle security concerns in networks. Much of the research surrounding new security mechanisms attempts to leverage new technologies, such as SDN, to try and bridge the gap of current security mechanisms and the requirements that they fail to satisfy. As more devices begin to connect to the internet, these concerns become more relative. With new solutions being developed and tested, the need for security will always fall behind in terms of innovation due to the rapid evolution of IoT devices. Overall, security will continue to rely on existing tools, techniques and practices to defend networks while new methods and mechanisms are developed to meet current and future security requirements.

REFERENCES

[1] Yawar Abbas Bangash, Yahya EA Al-Salhi, et al. 2017. Security Issues and Challenges in Wireless Sensor Networks: A Survey. *IAENG International Journal of Computer Science* 44, 2 (2017).

[2] Peter Bull, Ron Austin, Evgenii Popov, Mak Sharma, and Richard Watson. 2016. Flow based security for IoT devices using an SDN gateway. In *Future Internet of Things and Cloud (FiCloud), 2016 IEEE 4th International Conference on*. IEEE, 157–163.

[3] Olivier Flauzac, Carlos Gonzalez, Abdelhak Hachani, and Florent Nolot. 2015. SDN based architecture for IoT and improvement of the security. In *Advanced Information Networking and Applications Workshops (WAINA), 2015 IEEE 29th International Conference on*. IEEE, 688–693.

[4] Ben Greenstein, Damon McCoy, Jeffrey Pang, Tadayoshi Kohno, Srinivasan Seshan, and David Wetherall. 2008. Improving wireless privacy with an identifier-free link layer protocol. In *Proceedings of the 6th international conference on Mobile systems, applications, and services*. ACM, 40–53.

[5] Kubra Kalkan and Sherali Zeadally. 2017. Securing internet of things (iot) with software defined networking (sdn). *IEEE Communications Magazine* 99 (2017), 1–7.

[6] Aykut Karakaya and Sedat Akleylek. 2018. A survey on security threats and authentication approaches in wireless sensor networks. In *Digital Forensic and Security (ISDFS), 2018 6th International Symposium on*. IEEE, 1–4.

[7] Mehdi Nobakht, Vijay Sivaraman, and Roksana Boreli. 2016. A host-based intrusion detection and mitigation framework for smart home IoT using OpenFlow. In *Availability, Reliability and Security (ARES), 2016 11th International Conference on*. IEEE, 147–156.

[8] Ola Salman, Sarah Abdallah, Imad H Elhajj, Ali Chehab, and Ayman Kayssi. 2016. Identity-based authentication scheme for the internet of things. In *Computers and Communication (ISCC), 2016 IEEE Symposium on*. IEEE, 1109–1111.