24

A Review of Moving Target Defense Mechanisms for Internet of Things Applications

Nico Saputro[1,2], Samet Tonyali[3], Abdullah Aydeger[1], Kemal Akkaya[1], Mohammad A. Rahman[1], and Selcuk Uluagac[1]

[1] Department of Electrical and Computer Engineering, Florida International University, Miami, FL, USA
[2] Department of Electrical Engineering, Parahyangan Catholic University, Bandung, Indonesia
[3] Department of Electrical and Computer Engineering, Abdullah Gul University, Kayseri, Turkey

24.1 Introduction

In recent years, we have witnessed an exponential growth in the number of Internet of Things (IoT) devices around us [1]. IoT offers promising solutions in many application domains. Today, IoT devices and application are transforming the operations and role of many existing industrial systems from transportation to manufacturing to healthcare service systems. For example, in the transportation systems, IoT is used to realize Intelligent Transportation System (ITS) implementations [2]. There are also other on-going efforts to use IoT in critical infrastructures (CIs) for a nation such as in the transformation of the existing electric grid into the Smart Grid [3]. Similarly, their applications in the battlefield domain are also evolving [4]. Besides being used on the areas such as unmanned units (aerial or land or underwater) and troop health [5], there is a need to use IoT to support and augment critical-missions battlefield operations such as in the Intelligence, Surveillance, and Reconnaissance operations [6].

Security is a supreme importance for these applications. On the one hand, we can reap the benefits of the implemented IoT systems, yet on the other hand, they can pose significant security threats. However, securing them towards attacks from the

Modeling and Design of Secure Internet of Things, First Edition. Edited by Charles A. Kamhoua, Laurent L. Njilla, Alexander Kott and Sachin Shetty.

Published 2020 by John Wiley & Sons, Inc.

adversaries is very challenging. Typically, the applications and the used IoT devices are developed by a wide variety of organizations, either from for-profit organizations (e.g. start-up companies, small and medium enterprises, large corporations) or from non-profit organizations such as academic research institutions. Besides the interoperability issue between applications and devices that come from multiple developers, to keep pace with competitors, the developers often develop an IoT device only with the necessary network capabilities without any strong implementation of network security features in it. Furthermore, the security threats are augmented by the lack of physical security and upgradeability since these devices can be placed in any non-traditional locations such as in appliances, automobile, streets, or any remote locations. When the remote locations are inaccessible or very difficult to reach, upgrading is nearly impossible or too costly and thus the IoT devices are designed to operate for years with no means of long-term supports.

To secure an IoT system, one approach that can be done before the deployment is by incorporating security measures that address as many of the security vulnerabilities as possible at the design phase (a.k.a *security by design*). However, this approach may not enough. The lessons learned from securing the digital infrastructures, which have been studied for years before the imminent arrival of IoT, indicate that it is a never-ending battle between the defenders and attackers. Once an exploited vulnerability can be identified by the defenders and a patch and/or a defense mechanism against an attack that exploits this vulnerability is developed, attackers are able to find new vulnerabilities or increase their attack sophistication to outsmart the newly developed defense mechanism. This situation is driven by the fact that the existing network configurations of any digital infrastructures (e.g. network address, operating system, software, hardware, etc.) mostly remain static all the time. Thus, attackers have no time constraints of finding any vulnerabilities in the systems and exploiting these vulnerabilities at any time. Moreover, since other similar systems may use the same network components (e.g. operating system, vulnerable software), attackers can easily replicate their attacks to these systems.

Given these attackers' advantages, the defender's traditional approaches [7] that strive to directly deal with the vulnerability are no longer effective since they always fall-behind the attackers. Thus, complementary proactive self-defending approaches to break these advantages by introducing uncertainty through randomization of system attributes have been initiated in 2003. These types of approaches are later classified under the name of Moving Target Defense (MTD). MTD dynamically shifts the systems attack surface (e.g. IP addresses and ports, software/hardware vulnerability, memory location to store instructions and data, etc.) by leveraging redundancy (e.g. adding extra components with the similar functions) and diversity in the systems to make the attack surface unpredictable for attackers. Since the coined MTD in 2009, a variety of MTD techniques have been proposed and can be classified into five major categories based on their implementation on the software stack model [8]: (i) dynamic data; (ii) dynamic software; (iii) dynamic runtime environment; (iv) dynamic platform; and (v) dynamic network.

MTD techniques have been promising in many of the traditional network domains and thus such success has given rise to bring this experience to the IoT domain. Thus, in recent years, we have started to see studies that apply a variety of MTD techniques to numerous IoT environments (e.g. see Chapters 10 and 18 in this book) while there is still a lot of challenges yet to be addressed. Given the wide variety of MTD techniques, and the IoT characteristics, limitation, and the security and privacy issues in IoT, it is imperative for professionals and researchers working in IoT security to have a firm grasp on the available MTD techniques that are appropriate for the IoT systems depending on the application's needs.

Therefore, this chapter aims to provide a review of MTD approaches that revolve around IoT applications and then investigate the feasibility and potential of specific MTD approaches that will benefit some of the IoT applications the most. Our goal is to not only provide a categorization of various MTD approaches but to also lay down new research directions by advocating certain MTD techniques that can be used in conjunction with some of the emerging networking paradigms such as

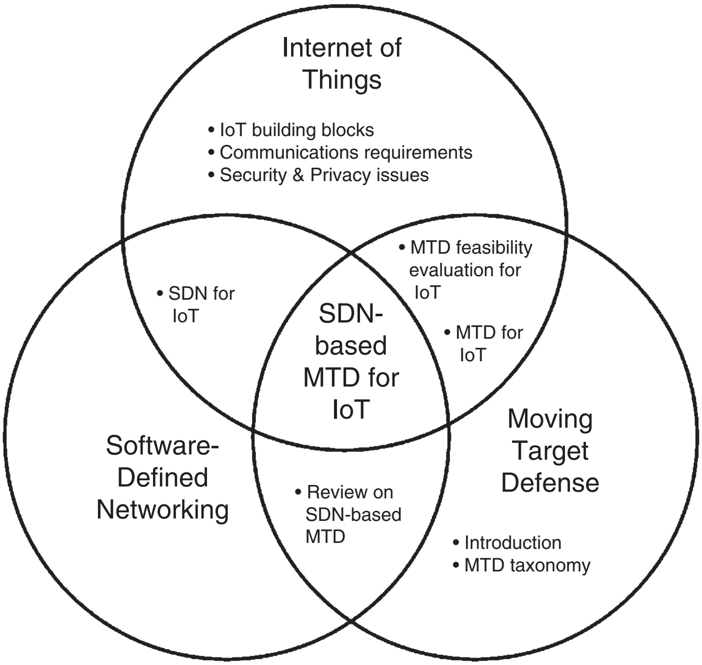

Internet of Things
• IoT building blocks
• Communications requirements
• Security & Privacy issues
• SDN for IoT
• MTD feasibility evaluation for IoT
• MTD for IoT
SDN-based MTD for IoT
Software-Defined Networking
• Review on SDN-based MTD
Moving Target Defense
• Introduction
• MTD taxonomy

Figure 24.1 Three-different domains – IoT, SDN, and MTD.

Software Defined Networking (SDN) [9]. This creates a rich intersection of multiple concepts as shown in Figure 24.1 that would benefit IoT security and defense.

The chapter begins the discussion by first laying out the foundation of MTD in the digital infrastructures. The discussion includes the motivation behind MTD and a brief overview of the existing MTD classifications. Then the security and privacy challenges for IoT as well as its characteristics and limitation are discussed. Based on that, we provide a brief evaluation of the feasibility of implementing each of the five major MTD categories, which are basically intended for the enterprise network, for IoT. For example, the dynamic platform that provides diversity by utilizing multiple operating systems, different processor architecture, storage systems, etc. may not be applicable in IoT system since it may be too costly to provide multiple processor architectures or storage systems when the role of the IoT device is merely for a simple task. Based on the feasibility analysis discussion, we then focus on the use of the dynamic network option for IoT.

In the dynamic network domain, the primary attempt for diversity is by modifying the network properties. In this chapter, we will first propose a taxonomy for the MTD techniques in the dynamic network domain. This categorization will be based on certain criteria in terms of techniques, network architecture, and the types of network attacks that these techniques strive to deal with. We then emphasize the MTD techniques that are supported by SDN paradigm which provides flexibility in terms of network management and control. A brief introduction on SDN will be provided prior to the discussion of the SDN-based MTDs.

After the discussion on the SDN-based MTD for the general digital infrastructures, a review on the existing works on MTD for IoT, either SDN-based or non-SDN based, is provided. The review follows a similar structure, i.e. it includes the techniques, network architecture, and the type of attacks. Finally, we provide future research directions and challenges that relate to SDN-based MTD for IoT. The chapter is finalized with a conclusion that summarizes the contributions.

24.2 Internet of Things

In this section, we first explain the IoT and the Industrial Internet of Things (IIoT) and its components. Then, we explain the communication requirements to build IoT and IIoT networks. Finally, we explain security and privacy concerns in IoT and IIoT applications.

24.2.1 Overview

The term IoT was initially referred to inter-operable objects with radio frequency identification (RFID) technology [10]. Then, its span expanded so as to cover other "things" that are capable of computing and communication as the market delivered low-cost, mobile computation and communication technologies. Today, the IoT is envisioned as a self-established global network infrastructure comprised of interconnected and uniquely identifiable physical artifacts, services, and humans that can be accessed from anywhere in the world through the Internet using standard communication protocols [11–15]. The IoT network aims to make these three parties communicate, and exchange data and information without human intervention as far as possible to fulfill a common need in different application domains [16, 17]. To this end, it enables Human-to-Thing, Thing-to-Thing, and Thing-to-Things communications [18]. The term of "thing" in the IoT concept mostly refers to compact smart devices [19] such as smartphones, tablets, digital cameras, automation systems, and controllers rather than typical computational platforms such as personal computers and workstations. Moreover, the things that are already a part of our environments such as home appliances, light bulbs, consumer electronics equipped with sensors, actuators that have communication interface, our cars, and offices and some other devices equipped with, say RFID tags, connected to the Internet through a gateway take part in the IoT.

Over the past decade, the number of connected devices has exceeded 15 billion. If this trend continues, it is expected that an estimate of 50 billion devices will be connected by the year 2020 [20]. Considering the drastically increasing number of connected devices, it can be deduced that the IoT has initiated a technological revolution in ubiquitous connectivity, computing, and communications.

The variety in the "things" that can take part in the IoT leads to a large number of application domains for the IoT. The domains are including but not limited to transportation, agriculture, food processing, healthcare, entertainment, home automation, industrial automation, surveillance and military, smart energy monitoring and management, and smart cities [12, 16, 21].

The IoT applications facilitate monitoring, controlling, and thereby interacting with the surrounding environment to make it more plausible. For example, smart cars, roadside units, and smart traffic signals make the driving experience safer and more convenient [16]. Another IoT-enabled technology is the ongoing Smart Grid initiative which enables high-frequency data collection compared to existing metering systems from the consumers, distribution substations, and transmission lines [22–26]. Such industrial systems that interconnect production systems and integrate them into conventional business information technology (IT) systems by incorporating the IoT technologies are called Industrial IoT (IIoT) [27].

The IIoT is a gateway towards digitizing the entire industry. Roughly speaking, this is achieved by collecting sensor data from industrial systems and transmitting them to more powerful computation and storage units over the Internet to analyze the system state and take convenient actions thanks to intelligent machine applications. This paves the way for more efficient production systems/manufacturing processes which are a key to the growth and competitiveness in a global economy [28] since the IIoT has the potential to improve quality control, supply chain management, sustainable services, and maintenance services for the user in the loop [29].

24.2.2 IoT Building Blocks

In this subsection, we explain the fundamental components of an IoT infrastructure.

An IoT ecosystem is typically composed of physical and digital worlds and the communication network that connects these worlds to each other [30]. As shown in Figure 24.2, the IoT architecture is a four-layer model comprised of IoT devices and gateway, communication network, cloud server, and IoT application.

IoT devices are equipped with sensors, actuators, controllers/processors, network interfaces, and short- and long-term memory. An IoT device runs its hardware through a firmware or an operating system that resides on its long-term memory. The device senses its surrounding environment through its sensors, stores the digitized data in its short-term memory to process the obtained data through its controller or processor, and communicates with an IoT application running on a cloud server which can be accessed over the Internet. The application processes the data collected from the IoT device and provides feedback to the device. Accordingly, the device takes an appropriate action ranging from displaying the result of a computation to switching a relay if it is a part of a cyber-physical system.

The IoT devices exchange data with other devices in the network or cloud servers to fulfill an IoT application's objective. The communication technology may differ while conveying the data towards the target device/cloud server which requires a protocol conversion because the communication network of an IoT ecosystem is the typical internet network having different layers (physical, link, network, transport, and application) with different protocols operating at each layer. In such cases, a gateway is installed where the protocol conversion is needed.

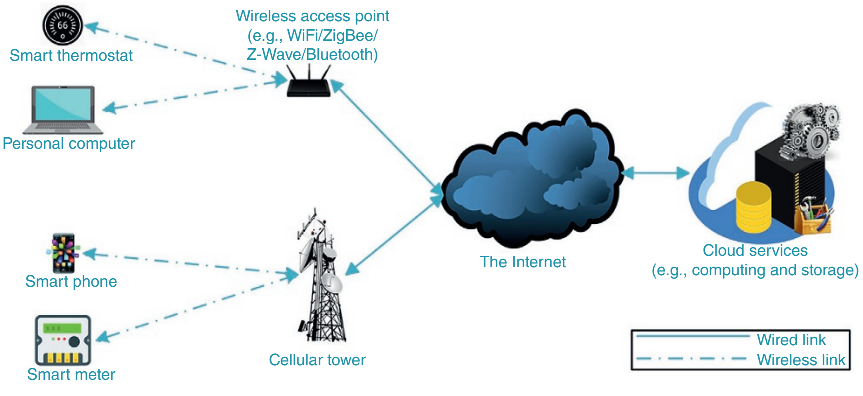


Figure 24.2 Essential building blocks of an IoT environment.

A gateway is supposed to be able to perform a bidirectional packet format conversion. For example, smart home appliances typically have a ZigBee [31] interface (Z-Wave [32] is also another promising protocol stack for smart home applications) and thereby communicate over the ZigBee protocol. To report the collected data from home appliances to the cloud server, the devices need a gateway that converts ZigBee packets into Long-Term Evolution (LTE) [33] or TCP/IP [34] packets and communicate with the cloud server by following the relevant protocol. Similarly, when the gateway receives some packets from the cloud server for the home appliances, it converts LTE or TCP/IP packets into ZigBee packets and follows the ZigBee protocol.

As mentioned above, the IoT devices communicate with a cloud server to store the collected data. These systems reside at the edge of the IoT system and have abundant storage resources and powerful processors. Thus, the stored data is used in data mining and analysis to make useful inferences for an IoT application. They also monitor the connected devices and manage device-to-device communication. They operate and synchronize different IoT devices and enable IoT applications. In addition, they communicate with other private and public servers or cloud services when needed for an IoT application.

An IoT application is the essential and indispensable part of an IoT ecosystem. It is a piece of software running on the cloud server that pre-processes, mines, and analyzes the stored data to derive useful insights and to manipulate the targeted IoT device(s) securely based on these insights. For instance, an IoT application designed for smart home automation can process data received from pressure sensors when households are not home, send commands from the cloud to lock the doors and windows and report the situation to the households and police department.

24.2.3 Communication Requirements of IoT

According to the statistics given in [35], there are approximately 25 billion IoT devices connected worldwide. The great majority of these devices are mobile, thereby should be equipped with required technology to be able to access the Internet whenever and wherever needed. Hence, it is inevitable to take advantage of wireless communication technologies because wireless solutions require far less cabling work and lower the infrastructure, deployment, and maintenance costs. Some of the wireless radio technologies currently used for IoT include 802.16 (WiMAX) [36], 802.11 (Wi-Fi) [37], and 802.15 (Bluetooth [38], ZigBee, and Z-Wave). These technologies differ from each other in the underlying PHY/MAC implementation which specifies the data rate, bandwidth, communication range, and so on. Hence, each of these technologies should be employed in convenient environments in which they can operate efficiently.

As we mentioned in Section 24.2.1, there will be more than 50 billion IoT devices connected by 2020. According to the survey conducted by SANS Institute [39], 72% of the survey participants who are industrial organizations base their communication infrastructure on Internet Protocol Suites where IP addresses are used to identify each device in IoT networks. Considering that there are fewer than 4.3 billion Internet Protocol version 4 (IPv4) [40] addresses (4 294 967 296 IPv4 addresses [41]) which are about to be exhausted, we can deduce that we will need a new mechanism to address the additional devices. Fortunately, Internet Protocol version 6 (IPv6) [42] has a potential of almost unlimited address space for more than trillions of devices [12]. Therefore, network stack of IoT devices is designed to support IPv6 routing protocol.

The IPv6 introduces an overhead such that the header occupies more space than the payload in a data fragment, which is not convenient for low-power and resource-constrained devices because this increases the number of datagrams to be transmitted. To overcome this problem, IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) protocol [43] was developed. 6LoWPAN is a low power communication network which connects resource-constrained wireless devices using compressed IPv6. It defines IPv6 header compression and how packets are routed through the IEEE 802.15.4 [44] links. It also defines fragmentation of IPv6 datagrams when the size is more than the IEEE 802.15.4 Maximum Transmission Unit (MTU) [45] which is 127 bytes.

An advantage of 6LoWPAN networks is that they support multihop communication (mesh network) where the nodes between a source and a destination node can forward data packets towards the destination node on behalf of the source node. Another advantage is on energy consumption. Instead of idle listening which is mostly used by power supplied devices, 6LoWPAN networks follow duty cycling paradigm which means that the radio is turned on only for a very short time for listening.

Currently used connection-oriented web protocols such as Hypertext Transfer Protocol (HTTP) [46] or HTTP Secure (HTTPs) [47] are designed to be used over Transmission Control Protocol (TCP) [48] which is not feasible for low-power and lossy links where it is hard to maintain a continuous connection between devices. Therefore, the Constrained Application Protocol (CoAP) [49] running on top of the connectionless User Datagram Protocol (UDP) [50] was developed for the IoT communication [12].

The protocols that were developed for the IoT communications can be listed as follows. Some of the most commonly used application layer protocols are CoAP, COAP Secure (COAPs) [49], Representational State Transfer (REST) [51], Message Queuing Telemetry Transport for Sensor Networks (MQTT-SN) [52], Live Long and Process (LLAP) [53], Extensible Messaging and Presence Protocol Internet of Things (XMPP-IoT) [54], and Advanced Message Queuing Protocol (AMQP)

[55]. TCP, UDP, Datagram Transport Layer Security (DTLS) [56], and Transport Layer Security (TLS) [57] are some examples of transport layer protocols. As mentioned above, IPv4 and IPv6 are the two most common network layer protocols. In addition, the Routing Protocol for Low-Power and Lossy Networks (RPL) [58] is a standardized routing protocol for the IoT. It is primarily designed for 6LoWPAN networks which are low-power and lossy networks. There is a wide variety of physical or link layer protocols that can be used for IoT. Low-Rate Wireless Personal Area Network (LR-WPAN) [59], Bluetooth/Bluetooth Low Energy (BLE) [60], 802.15.4, LTE, General Packet Radio Services (GPRS) [61], Collision Detect Multiple Access (CDMA) [62], Carrier Sense Multiple Access (CSMA) [63], CSMA Collision Detection (CSMA/CD) [64], CSMA Collision Notification (CSMA/CN) [65], Zigbee, 802.11 Wi-Fi, Near Field Communication (NFC) [66], Wireless Highway Addressable Remote Transducer (WirelessHART) [67], Z-Wave, Sigfox [68], DASH7 Alliance Protocol (D7A) [69], Long Range Wide Area Network (LoRaWAN) [70], Thread [71], and INSTEON [72] are some of the most known physical/link layer protocols [30]. In Table 24.1, we provide the IoT technologies developed for each OSI layer and their implementations in the Contiki operating system [12].

When it comes to IIoT both wired and wireless communication technologies can be used. In addition to those used by IoT, Ethernet, Profinet [81], and ModBus [82] are some examples to the protocols developed particularly for IIoT. MQTT was designed for IIoT networks although it is also used by IoT devices. It is a light-weight publish–subscribe messaging protocol which enables two-way machine-to-machine communications [14, 83]. Another publish–subscribe-based machine-to-machine messaging protocol is Data Distribution Service [84] which has a wide variety of application domains such as autonomous vehicles, air-traffic control, smart grid, robotics, medical devices, and so on.

Table 24.1 IoT technologies developed for each OSI layer and their implementations in Contiki OS.

OSI layer	IoT technology	Contiki implementation
Application	CoAP, CoAPs	Erbium [73]
Session	DTLS	TinyDTLS [74]
Transport	UDP	*micro*IP [75]
Network	IPv6, RPL, IPSec [76], 6LoWPAN	*micro*IP, ContikiRPL [77], IPSec in Contiki, SICSLoWPAN [78]
Data link	802.15.4 MAC	ContikiMAC [79]
Physical	802.15.4 PHY	Contiki 802.15.4 [80]

24.2.4 Security and Privacy Issues in IoT

Security has been an indispensable component of computers and computer networks. In a similar manner, IoT devices and IoT networks are expected to meet some security requirements. Hence, IoT security is an emerging research topic that is attracting attention both in academics and industrial sector. There are plenty of international organizations and companies that focus on the design and development of IoT-based systems. However, commercialization of IoT resulted in an increase in public security concerns such as privacy issues, cyber-attacks, and organized crime [19].

The integration of smart devices into the Internet introduces several security problems due to two major reasons. The first reason is the heterogeneity in the IoT devices. This causes some inconsistencies among the devices especially in security mechanisms employed because some of the devices are resource-constrained devices while the others are powerful hosts. The second reason is that most of the Internet technologies and communication protocols were designed for traditional web communications, but not to support IoT. This has led to the development and standardization of IoT-specific security protocols (e.g. lightweight compressed IPsec, lightweight DTLS, IEEE 802.15.4 link-layer security, etc.) that ensure end-to-end message integrity and confidentiality [16].

IoT devices are exposed to cyber threats from the Internet as well as from the local network they are connected to. Thanks to Shodan [85], it has been very easy to find some specific types of IoT devices linked to the Internet such as IP cameras, routers, servers, and so on. Shodan helps to find the public IP address of these kinds of devices. These devices are delivered with default username and password to access their management interface. Most of the owners do not change the username and password, and this enables cyber attackers to compromise the devices and use them for their cruel purposes. Mirai botnet attack [86] is one of such attacks taken place recently. Therefore, to thwart botnet attacks, California signed a new bill into law called "Security of connected devices." According to the law, device manufacturers are going to include either a pre-programmed password unique to each device or a mechanism that enables the user to generate a new means of authentication before being granted access to the device for the first time [87].

Security goals can be implemented with some policies guided by the confidentiality, integrity, and availability (CIA) triad [88]. Confidentiality is the first thing that comes to mind when it comes to security because it is not reasonable to envision a secure system in which sensitive data is stored in plaintext. Some data perturbation methods such as data obfuscation, encryption, and secret sharing are used to conceal users' sensitive data. Maintaining integrity of data at rest, code running on the device or a network packet is another essential item in the triad.

For instance, it is extremely important to ensure that a packet is not tampered with during its journey from the source to the destination. Message authentication codes and digital signatures are the two most commonly used techniques to prevent adversaries from breaking the integrity of data or message. Moreover, the data at rest local/remote or a web service should be reliably accessible by authorized parties whenever they attempt to access. This mostly depends on the performance of storage devices, so maintenance of hardware and design of software that makes the data/services available. A robust and resilient system avoids creating bottlenecks, provides adequate communication bandwidth and fast and adaptive disaster recovery.

As might be expected, IIoT also faces similar security challenges. One of the most significant security challenges in IIoT is to assure CIA-compliant but at the same time traceable and transparent data exchange between multiple stakeholders [89]. To comply with CIA features, certificates can be maintained, but it is time-consuming to use them due to the significant management overhead for certificate storage, distribution, verification, and revocation in traditional PKI [15]. Therefore, most of the IT/operational technology (OT) department managers would prefer network segmentation using firewalls, data diodes (unidirectional gateways), and IT/OT gateways to protect manufacturing systems against the risks imposed by existing and new IIoT devices [39].

The introduction of Internet technologies into the industrial domain drastically enlarges the potential attack surface and, therefore, poses some security risks. In a probable attack scenario, remote attackers can intrude themselves into an industrial communication network and gain privileged access to the equipment's processors although they do not have physical access to the hardware of the devices. They can get access to the authentication mechanisms and extract authentication keys after a successful attack. Moreover, remote attackers may have local partners inside the facilities which are called insider attacker. Insider attackers have physical access to the equipment and make it easy to obtain the credentials from the equipment's memory [90].

To reduce attack surface and protect their networks, industrial IT/OT security specialists have isolated their systems in cyber-domain. However, such an isolation cannot thwart zero-day vulnerability exploits and insider attacks. The attacks against power and nuclear plants performed by Stuxnet worm, Havex and Black-Energy trojans, EternalBlue and WannaCry attacks, and Dragonfly group are major examples that demonstrate that isolation is not a panacea for security concerns of industrial control systems [91].

We have focused on security concerns so far. In addition to security concerns, frequent data collection from IoT devices puts the users' privacy in risk as it can help expose the users' location and daily habits to the service providers. For example, collection and storage of fine-grained energy consumption data raise the issue of privacy for the consumers who must use smart meters daily. Specifically, the collected consumption data can be analyzed using load monitoring techniques to infer activities of the consumers. Hence, typical privacy threats include, but not limited to: (i) Determining personal behavior patterns (can be used by marketers, government); (ii) Determining specific appliances used (can be used by insurance companies); (iii) Performing real-time surveillance (can be used by law enforcement and press); (iv) Target home invasions (can be used by criminals); and (v) Location tracking based on electric vehicle usage patterns (can be used by law enforcement). Secure in-network data aggregation can be used to both preserve consumers' privacy and reduce the packet traffic due to high-frequency metering data. The privacy can be provided by performing the aggregation on concealed metering data. Fully homomorphic encryption and secure multiparty computation are the other two systems that enable performing multiple operations on concealed data [24].

24.3 Software Defined Networking Background

IoT devices are both connected to each other and to the Internet, which forms a large-scale network. The communication between each other and to the Internet is established through the routers/switches deployed on the network. These devices are usually expensive and run the complex proprietary software. The software running on these devices is also vendor specific and each device should be configured individually by the network admin. Considering IoT devices and massive network communication between them, managing the network devices individually is very challenging since there can be too many of them in a small environment, such as in a smart home.

Meanwhile, researchers proposed SDN recently which is an emerging technology providing great flexibility and cost-effective solution to the control and management of the networks [9]. SDN proposes a separation of data and control planes and that is the main difference from the traditional network environment as shown in Figure 24.3. On the one hand, the data plane consists of switches which are not capable of any routing/blocking decisions by themselves. On the other hand, the control plane (also known as SDN Controller) is the brain of the network and is responsible for all critical operations in the network. SDN provides a flexible and cost-effective solution for network management by enriching network devices with programmability feature and a centralized controller can be used by the network admin to configure the network devices. Even though network programming has been investigated for a long time, SDN has been improved and named since a couple of years now [9]. With the introduction of SDN, it is made possible to do

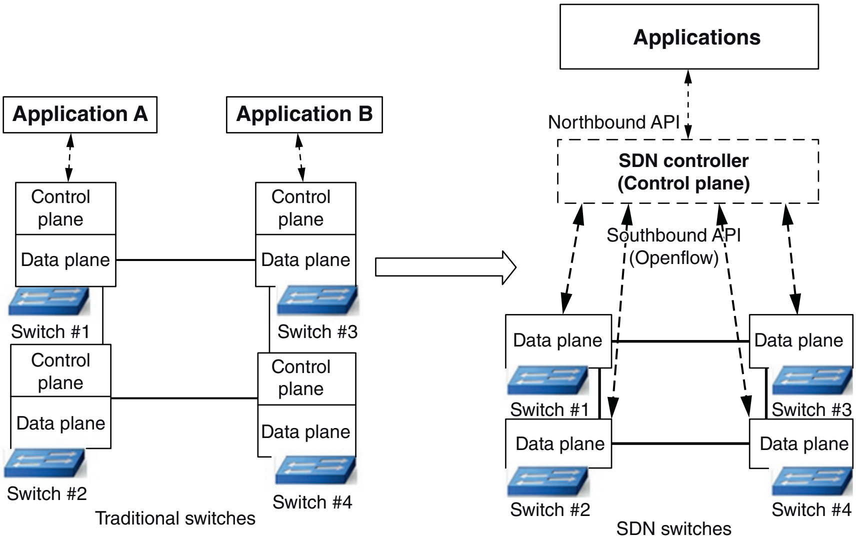


Figure 24.3 Traditional network switches versus SDN-based switches.

dynamic traffic engineering, drop packets, reconfigure the links in case of failures and enforce certain policies which make network management convenient and more flexible.

Traditional network devices (switch, router, etc.) require all the software on the hardware to run protocols before it can be installed in the network. Meanwhile, SDN-based switches are basic devices and can be updated easily even after the installation. However, there is a need for a new communication protocol for SDN Controller to communicate to SDN switches. Even though there is not a standardized protocol yet, OpenFlow is the widely-used protocol by both researchers and industry [92]. SDN-based switches are also called OpenFlow switches and these switches have flow tables that will provide the knowledge of the rules on what to do with each packet. *Flow table*, *OpenFlow Protocol* and *secure channel* between SDN Controller to OpenFlow switches are the main parts of OpenFlow switches. In addition to the mentioned OpenFlow switches and the protocol requirements, there is also a need to run an SDN Controller which network administrators can use to access the network devices remotely. For example, FloodLight [93] and OpenDayLight [94] are publicly available and preferred by network admins.

The SDN Controller typically should be running on a different machine in a centralized location and only network administrator(s) should have permission to access it. The SDN Controller can do many operations including but not limited to enforce security rules (block some packets, etc.) and decide forwarding tables. By exposing SDN-based solution to networks, the high cost of network devices and complexities of maintenance of such devices can be minimized. The network administrator can configure and update some parts of the SDN Controller to manipulate the network or s/he can implement some applications on top of SDN Controller to apply his/her own rules. The latter is more common and requires less knowledge of SDN Controller's source code. Northbound API of SDN Controller is required to be used for this purpose and mostly used protocol is REST interfaces [95]. REST interface is turning network devices to Web applications. These applications send REST inquiries to get information about the current situation or to update some flow tables by using SDN Controller interfaces.

24.4 Moving Target Defense

24.4.1 Introduction

In 2009, five new game-changing directions have been introduced to address some intractable problems in the digital infrastructures [96]. These directions strive to solve these problems from a different perspective instead of the typical traditional approaches that tackle them directly. One of these directions called MTD, was driven by the fact that the digital infrastructure settings are relatively static for a long period of time. For example, once the computer that we use every day has been installed with an operating system (OS) and assigned an IP address, these settings are barely changed. These conditions enable attackers to have unlimited time to perform any stage of the five-phase attack kill chain [8]. MTD enables defenders to build proactive self-defense mechanisms that dynamically change the digital system attributes while still ensuring the system accessibility for legitimate users. These self-defense mechanisms introduce the redundancy and diversity in the system to make the attack surface unpredictable for attackers. For instance, instead of using a single OS platform all the time, a computer can dynamically rotate its OS to a different OS platform at a random time interval. This way, it will make it harder for attackers to perform their attacks since their insight of the digital system from their previous attack attempt may become obsolete since the OS platform has changed.

24.4.2 A Brief Review on MTD Classifications

A variety of MTD techniques have been proposed since the MTD is coined in 2009. They can be classified into five top-level categories [8]: (i) *dynamic data*, which covers MTD techniques that dynamically change the data representation properties (e.g. format, syntax, encoding, etc.); (ii) *dynamic software*, which covers MTD techniques that utilize multiple equivalent functions to provide an application diversity by dynamically changing the application codes; (iii) *dynamic runtime environment*, which covers MTD techniques that strive to modify the operating system (OS) to provide the diversity, either through address space randomization (ASR) or instruction set randomization (ISR); (iv) *dynamic platforms*, which covers MTD techniques that utilize multiple unmodified OSes and other platform characteristics (e.g. processor architectures, virtual memory, storage systems, etc.) to enable the platform diversity by dynamically migrating between platforms; and (v) *dynamic network*, which covers MTD techniques that dynamically modify the network properties (e.g. addresses, ports, routing, etc.). Each category attempts to address a different attack phase of the five-phase attack kill chain (i.e. reconnaissance, access, development, launch, and persistence phases). The dynamic data, software, and runtime environment domains mainly focus on the *development* and *launch* phases. The dynamic platforms domain attempts to disrupt any attacks at the *access* and *persistence* phases while the dynamic network domain attempt to address the *reconnaissance* and *launch* phases.

Specifically in the dynamic network domain, the existing MTD techniques can be further classified into four categories: (i) *dynamic network-identity*, which covers any attempts that strive to introduce the dynamic to the network identity, such as the physical address, logical address, and port number [97–103]; (ii) *dynamic network-path*, which covers any attempts to provide the dynamic paths with the intention to thwart any network attacks [104,105]; (iii) *dynamic network-infrastructure*, which covers any attempts to introduce the dynamic to the network components such as by dynamically changing the proxies [106–108]; and (iv) *dynamic network traffic configuration*, which covers any attempts that strive to provide the dynamicity to the network traffic such as the dynamic time schedule for periodic traffic, data size, and the dynamic protocol information with the aim of obfuscating the attackers [109, 110]. Figure 24.4 shows our proposed MTD classification.

Among these categories, most of the research efforts fall into the dynamic network-identity category since it can be done on the legacy network with the minimum to moderate legacy network modifications. However, this is not the case for the dynamic network-path on the legacy network. Typically, network paths are static when there is no performance of failures issues [105]. Moreover, orchestrating dynamic paths among network devices is very challenging since most routing protocols are distributed in nature. Every network node operates based on the local knowledge to determine the path for a traffic while the dynamic network-path MTD requires a coordinated effort that needs a global knowledge to provide the path diversity. Therefore, a completely new dynamic routing protocol may be needed to support the dynamic network paths randomization. For the last two categories, the dynamic network-infrastructure, and dynamic network traffic

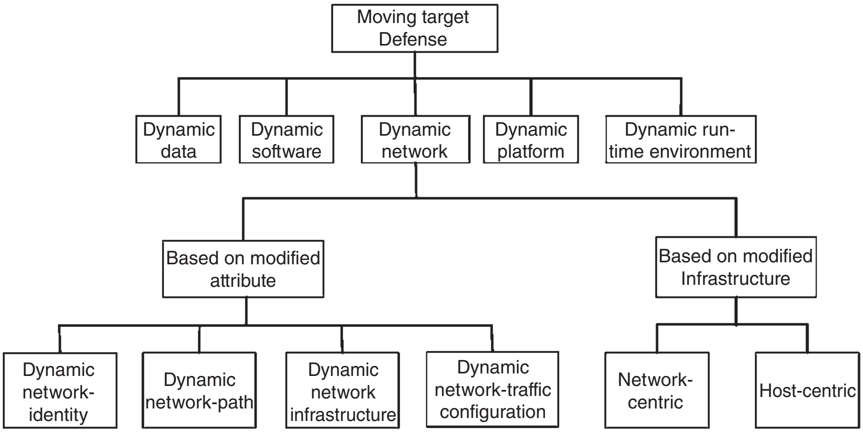

Moving target Defense
Dynamic data
Dynamic software
Dynamic network
Dynamic platform
Dynamic run-time environment
Based on modified attribute
Based on modified Infrastructure
Dynamic network-identity
Dynamic network-path
Dynamic network infrastructure
Dynamic network-traffic configuration
Network-centric
Host-centric

Figure 24.4 Proposed MTD classification for the dynamic network. This classification is extended from the MTD classification in [8].

configuration, on the other hand, (even though they can be implemented on the legacy networks), the possible moving options are somewhat limited and application-dependent (e.g. smart grid [109, 110]).

Based on which network infrastructure will be modified/added to support the MTD based network-identity randomization operations, an MTD approach can be classified as a host-centric [97, 102] or a network-centric approach [98–101]. In the host-centric MTD approach such as the TAP-based port and address hopping (TPAH) [97], the communicating end-hosts need to be modified to support the MTD operations. TPAH approach requires a hopping engine that needs to be added between the TAP virtual-network kernel driver and the physical network adapter as depicted in Figure 24.5. This hopping engine consists of two modules: (i) a port and address hopping module that handles port and address mutation and mapping; and (ii) an access control module that performs traffic monitoring, access control, and port and address mapping. The TAP driver acts as tunnel between a user-space process and the Operating System network stack. Operating System sends packet to a user-space process via a TAP driver and a user-space process can sends packets to the TAP device, which then injects these packets to the Operating System network. In the network-centric approach, network infrastructure needs to be modified such as by adding varying number of new gateways [98–101]. These gateways are typically located at the boundaries of the physical subnet to ensure the address translation between the real address and the short-lived temporary addresses. Figures 24.6 and 24.7 show an example of the network-centric architecture with three and two gateways respectively.

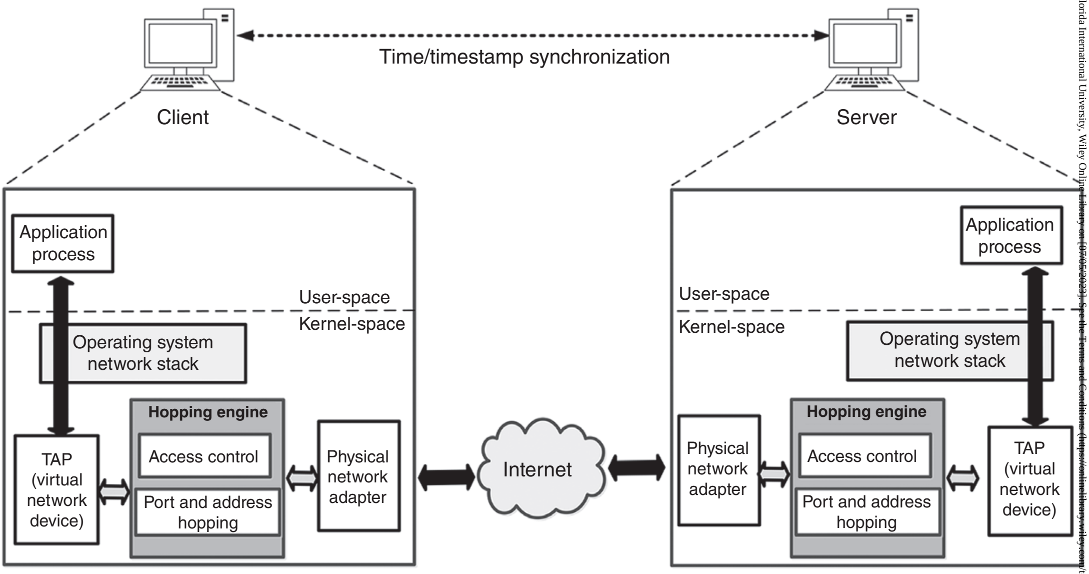


Figure 24.5 An example of host-centric approach: TAP-based port and address hopping (TPAH) architecture

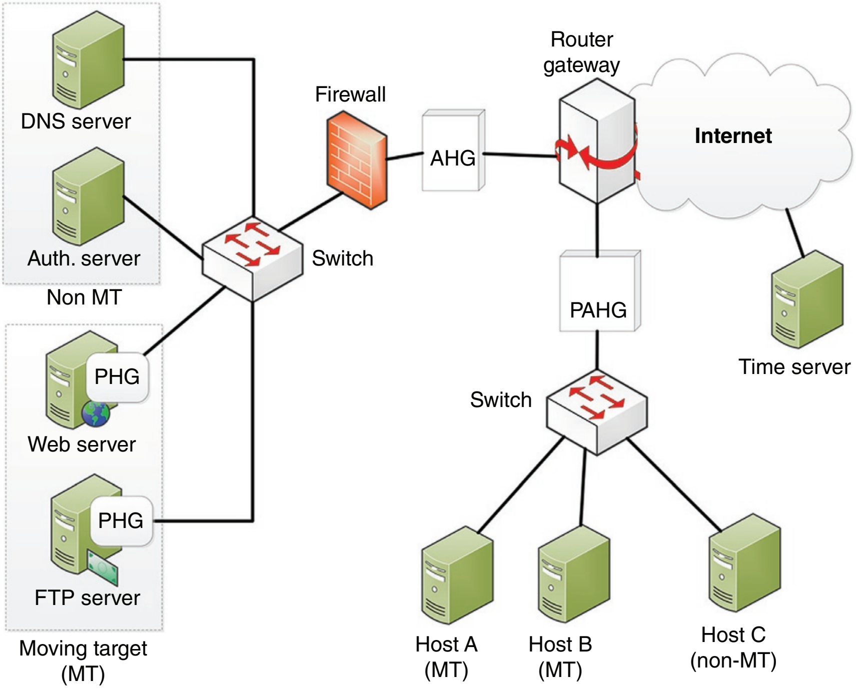


Figure 24.6 An example of network-centric architecture with three additional gateways, to support RPAH operations, adapted from [98]: Port Hopping engine Gateway (PHG), Address Hopping Gateway (AHG), and Port and Address Hopping Gateway (PAHG).

24.4.3 SDN-Based MTD Overview

The presence of emerging SDN technology that offers the separation of data and control plane, which in turn improves the network visibility and policy enforcement capability when compared to the legacy network, has brought the MTD research to the next level. There has been an increasing number of SDN-based MTD studies in the recent years, not only related to the MTD in the dynamic network domain [104, 111–124], but also to other areas such as for cloud networks [125, 126]. An overview of the SDN roles and the MTD techniques used to introduce the dynamicity in the networks is reviewed in the following subsections. The review is organized according to the dynamic network category (if applicable) in Section 24.4.2. Additionally, a separate section is provided for the review of the hybrid approaches, which utilize more than one dynamic network category collaboratively. Figure 24.8 shows the proposed SDN-based MTD classification.

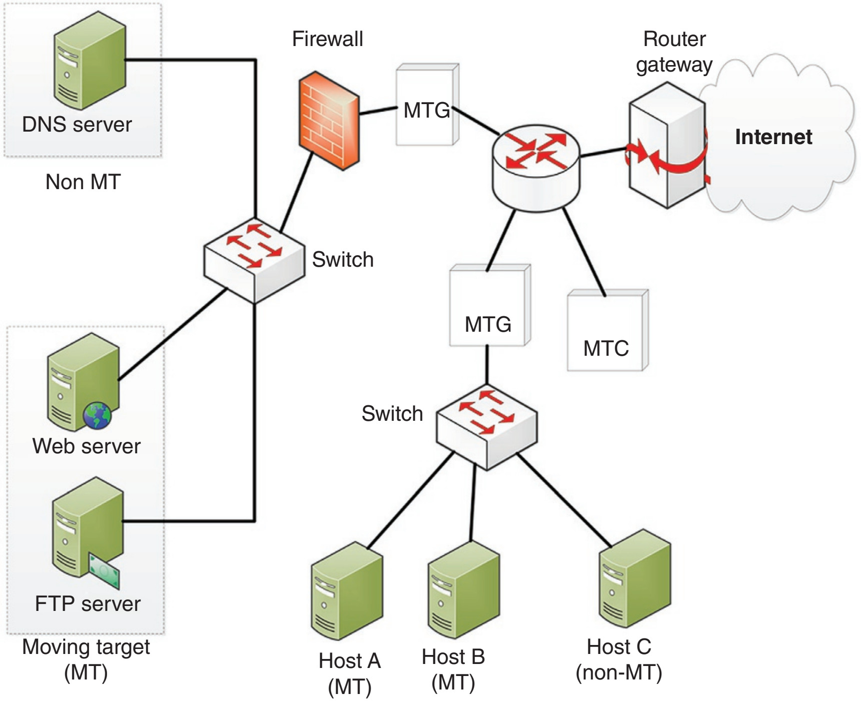


Figure 24.7 An example of network-centric architecture with two additional gateways to support the RHM operations: Moving Target Gateways (MTGs) and Moving Target central Controller (MTC).

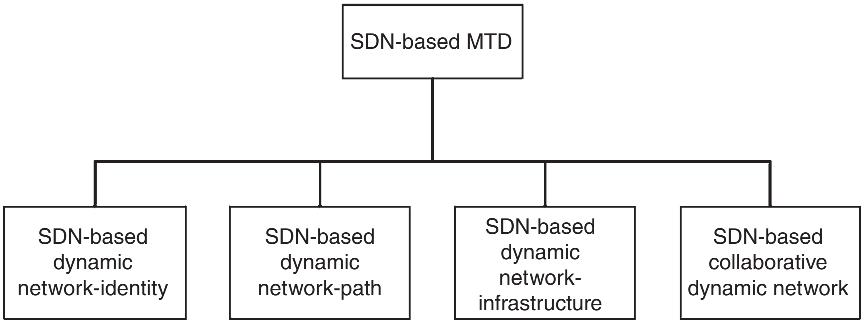

SDN-based MTD
SDN-based dynamic network-identity
SDN-based dynamic network-path
SDN-based dynamic network-infrastructure
SDN-based collaborative dynamic network

Figure 24.8 Proposed SDN-based MTD classification. This classification is derived from the existing works in the SDN-based MTD.

24.4.3.1 SDN-Based Dynamic Network-Identity

In the dynamic network-identity domain, instead of shuffling the real identity of a network device, a short-lived virtual identity is used in the communications between network devices to minimize the reconfiguration overhead on network hosts and to make the address mutation transparent to the end host. This short-lived virtual identity can have many different names depending on the approach, e.g. synthetic address [112], virtual address [99, 111], or ephemeral address [100, 101]. When SDN is employed in this domain, SDN can be used either in the host-centric based approach [112] or in the network-centric based approach [99–101, 111, 113].

The SDN shuffle technique [112] is a host-centric approach to defend against malicious reconnaissance. This technique enables both MTD clients and non MTD clients to access MTD (i.e. protected) servers by utilizing the SDN controller as the network address generator for the synthetic MAC and IP addresses. When the clients (either MTD or non-MTD clients) request a Domain Name Service (DNS) resolution of a protected server, the DNS server forwards the protected server synthetic IP address generated by the SDN controller to the client with a very low time-to-live (TTL) information to ensure that the client will reissue a new DNS resolution for each new connection. The corresponding synthetic MAC address for this synthetic IP address will be sent to the client in response to the ARP resolution of the synthetic IP address. Hence, both synthetic physical and logical addresses of the protected server are different for different clients. Figure 24.9 illustrates the SDN shuffle protocol operations.

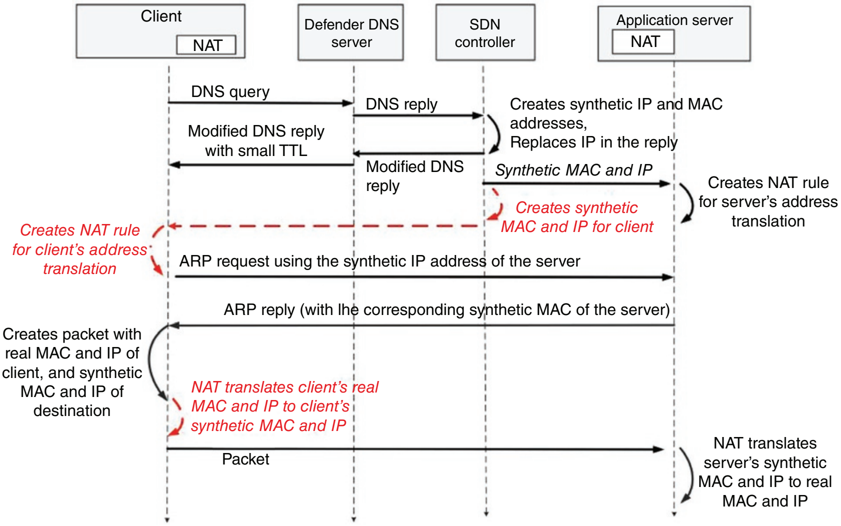

Client
NAT
Defender DNS server
SDN controller
Application server
NAT
DNS query
DNS reply
Creates synthetic IP and MAC addresses,
Replaces IP in the reply
Modified DNS reply with small TTL
Modified DNS reply
Synthetic MAC and IP
Creates NAT rule for server's address translation
Creates synthetic MAC and IP for client
Creates NAT rule for client's address translation
ARP request using the synthetic IP address of the server
ARP reply (with lhe corresponding synthetic MAC of the server)
Creates packet with real MAC and IP of client, and synthetic MAC and IP of destination
NAT translates client's real MAC and IP to client's synthetic MAC and IP
Packet
NAT translates server's synthetic MAC and IP to real MAC and IP

Figure 24.9 SDN shuffle protocol operations. The dotted lines are optional protocol operations when a client also employs the synthetic addresses.

In the network-centric approaches, SDN is used as the alternative replacement for the role of the MTD gateways in the legacy networks. For example, Figure 24.10 illustrates the RHM operations when it is used in the legacy networks and in the SDN-based network. SDN switch performs the address translation mechanism in the SDN-based network.

Packet Header Randomization (PHEAR) Technique [113], is a network-centric anonymous SDN-based communications system for enterprise networks that provide traffic unlinkability from its communicating hosts. It creates an identifier free traffic by removing both the explicit (e.g. MAC and IP address) and implicit identifiers (e.g. the IP initial TTL and TCP initial window size information may reveal

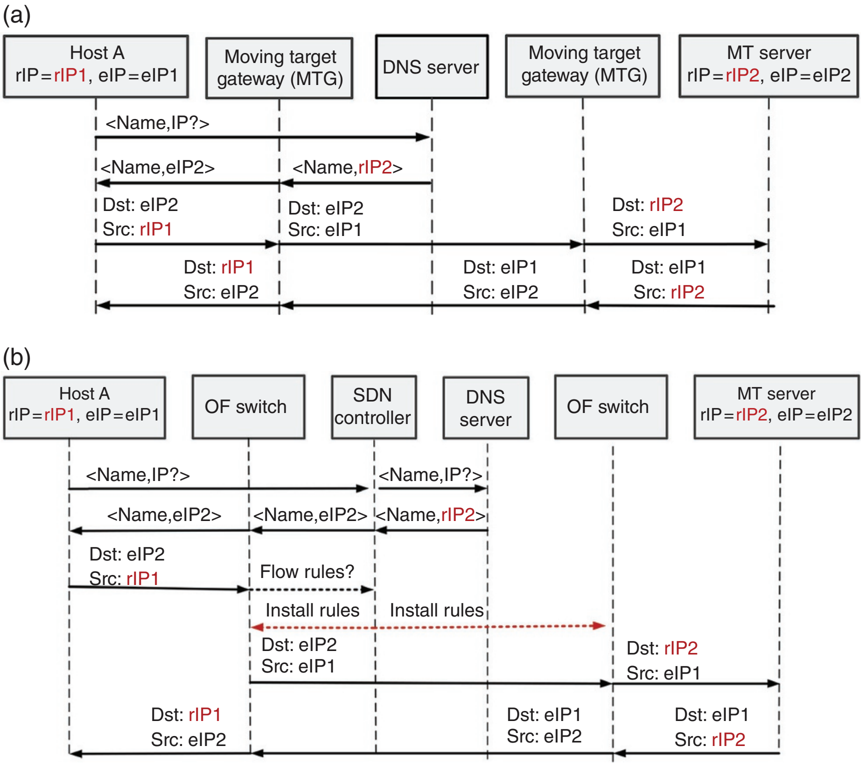


Figure 24.10 RHM operations comparison when a client tries to access the server using the domain name. In (a), gateways are used on the legacy network while in (b), an SDN-based network is used. When the MTG and SDN controller intercepts the DNS response on the domain name request, besides translating the real IP (*rIP*) to the ephemeral IP (*eIP*), they also change the time-to-live (TTL) to ensure that network hosts will perform DNS request again for each new connection.

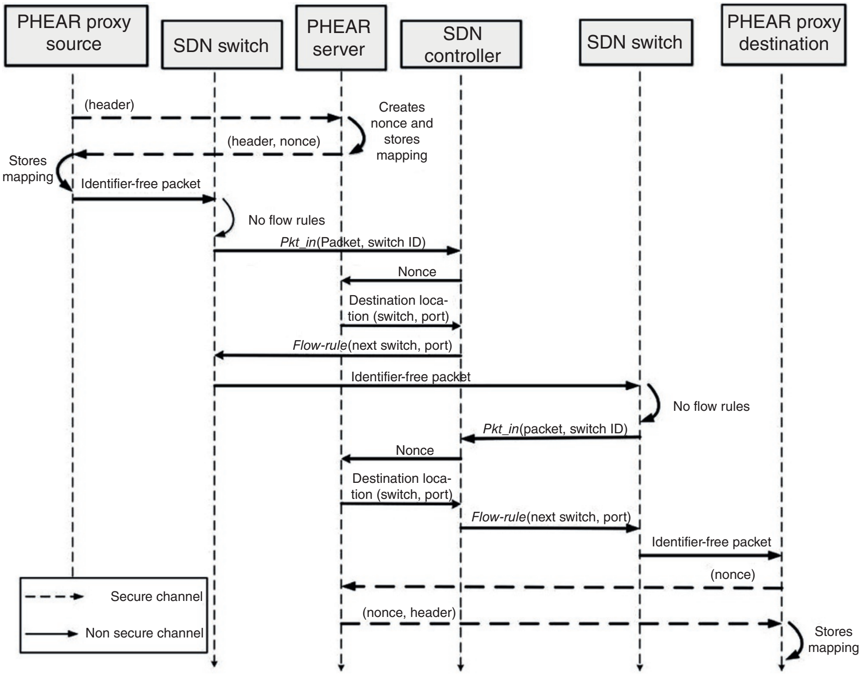


Figure 24.11 PHEAR per-switch basis operations.

the Operating system platform) from the packet headers while hiding the transport layer and above using the IPSec protocol. A short-lived 64-bit nonce is used to replace the source and destination IP address fields in the packet headers.

PHEAR leverages the SDN for the identifier-free packet forwarding using this short-lived nonce. Two types of network components need to be added to the SDN network to support the PHEAR operations: (i) a PHEAR server that creates a unique nonce for each packet header using a collision-resistant hash function and maintains its mapping; and (ii) a PHEAR proxy that resides on end-hosts and is responsible for the translation. Figure 24.11 illustrates the PHEAR per-switch basis operations.

24.4.3.2 SDN-Based Dynamic Network-Path

The SDN programmable feature and centralized control through the separation of data and control plane, has enabled proactive defense MTD against reconnaissance, eavesdropping, and DoS attacks by dynamically changing the network path [104, 119–123]. SDN eliminates the tedious tasks that must be done when implementing dynamic network-path MTD in the legacy network with respect to automatic routing table update to support any route changes without interrupting the on-going traffic or violating any security requirements. With SDN, changing routes can be done as a series of flow table updates in the SDN switches. In contrast, updating a routing table in the legacy network can be done by issuing a static route command manually [104] and/or by using a dynamic routing protocol. Obviously, issuing static route command is not an option due to a scalability issue. As previously mentioned in Section 24.4.2, a new routing protocol may be needed to ensure a timely update that will not interrupt the on-going traffic. One way to solve this problem is by utilizing MPLS (multiprotocol label switching) [127], which requires MPLS routers to carry out the operations. MPLS utilizes labels to identify virtual paths. Data packets are assigned labels and the forwarding decisions are made solely based on the label. MPLS, however, requires establishing the virtual path through the bandwidth reservation mechanism.

In the SDN-based network, the SDN controller acts as the central coordinator of the route mutation. The difference between approaches is typically on the route selection criteria. The random route mutation (RRM) [104] selects the route that satisfies the capacity, overlap, and quality of service constraints. When more than one eligible routes are found, one route will be randomly selected. In [119], a security constraint, which considers any previously used access control policies, is added into RRM route selection criteria. The route mutation in [120] selects the route based on the overlap constraint criterion to defend against the reconnaissance phase of the Crossfire attack, an indirect distributed DOS (DDoS) attack. As the network size increases, the efficiency of the route mutation becomes an issue since the bigger the network, the longer the processing delay of the route mutation to find the route(s) that satisfy the constraint(s) [121]. To address this issue, an effective and faster route mutation approach called Area-dividing Random Route Mutation (ARRM) [121] approach is proposed. The idea is by dividing the entire network into sub-areas and a backbone area. The communication between areas is enabled through the backbone. This way, when the internal link states in an area are changed, the ARRM approach only needs to calculate the route mutation for that area.

24.4.3.3 SDN-Based Dynamic Network *Infrastructure*

In an SDN-based network, the SDN controller plays a key role in network operations. Yet, it also can be the single point of failure of the network. Thus, the SDN controller becomes the ultimate attack target to disable the network. Considering a new type of DDoS attack, called the Blind DDoS attack, an MTD approach that provides diversity through dynamically changing the active SDN controller from a pool of SDN controllers has been proposed [116]. This type of DDoS attack attempts to attack the SDN controller by flooding the SDN switch with many packets that cannot be processed. In this case, the switch will forward them to the controller. Hence, the controller's capability will be degraded when packets from multiple SDN switches are forwarded to the controller.

24.4.3.4 SDN-Based Collaborative Dynamic Network

Recently, a hybrid MTD approach that combines different dynamic network categories to provide a proactive defense has been proposed. This collaborative approach typically combines the *Dynamic Network-identity* with the *Dynamic Network-path* [122–124]. This approach is also known as the *double hopping* approach. The differences between approaches are related to the attack models and the considered constraints in the approaches. The considered attack model in the Path Hopping SDN Network Defense (PH-SND) [122] is the traffic analysis attack that attempts to intercept and examine traffic to deduce information from the traffic patterns. Besides selecting a route based on the capacity and overlap constraints, PH-SND modifies the address and port information of the source and destination nodes during the forwarding process to further confuse the traffic analysis. Similarly, the Double Hopping Communication (DHC) [123] also modifies the address and port information and selects a route based on the path-length and overlap constraints to address the sniffer attack.

Another double hopping approach, a self-adaptive End-point Hopping Technique (SEHT) [124] is proposed to address the lack of ability from the existing approaches to adapt to different attacks strategies. Only a few existing approaches are the adversary-aware approach, which considers any potential attackers' actions. For example, Jafarian et al. [101] studied the use of a fast and accurate hypothesis testing for characterizing the adversarial scanning strategies in the legacy networks and adapts accordingly. Similarly, SEHT approach also considers the scanning attacks as its attack model and utilizes an analysis engine to perceive and analyze the adversary attack strategy. Four constraints are considered in SEHT: capacity, reachability, forwarding path delay, and hopping space selection constraints.

24.5 Moving Target Defense for IoT

In this section, we first discuss the feasibility of MDT for IoT environments and then move on to review the existing works under various categories.

24.5.1 A Brief Evaluation of MTD Feasibility for IoT

Besides the typical unconstrained nodes (e.g. servers, desktop computers, and powerful mobile devices such as smartphones), a plethora of IoT end-nodes are constrained nodes that have limited resources such as limited computation power,

Table 24.2 Constrained device categories (RFC 7228).

Class	Memory	Storage	Remark
Class 0	<<10 KiB	<<100 KiB	Typically, battery-operated, not enough space for the full IP stack and security implementations, typically preconfigured with a very small data set, assisting by gateways for Internet communications
Class 1	~10 KiB	~100 KiB	Enough space for optimized protocols such as CoAP (Constrained Application Protocol), it does not need any gateways assistance for Internet communications
Class 2	~50 KiB	~250 KiB	Enough to support full IP stack implementation and can be fully integrated into IP networks

limited memory, limited storage capacity, and limited power capacity (e.g. battery-operated devices). The Internet Engineering Task Force (IETF) has defined three constrained node categories in RFC 7228 as presented in Table 24.2.

The inherent limitations in each class eventually will limit the applicability of any MTD domains in the IoT. The class-0 IoT devices are severely constrained nodes with very limited memory and storage capacities since they only need to do very simple tasks (e.g. send an on/off or other basic health indicators). These capacities are not even enough for the IP stack and security implementations. Hence, any MTD domains cannot be implemented on the class-0 devices due to these resource limitations. Instead, since the IoT devices in this class utilize gateways, which are typically unconstrained nodes, for Internet communications, MTD can be implemented at the gateways level. For the other two less-constrained classes (i.e. class-1 and class-2), the MTD techniques need to be designed and optimized by considering these limitations.

Among the five top-level categories of MTD techniques, the dynamic data domain may not be affected by the resource-constrained limitation, unless encryption mechanisms are involved in the data encoding. Most of the existing works are in the dynamic network domain as will be explained in the next section. The works on the other domains, however, are limited.

The maneuverability of the dynamic software and dynamic run-time environment domains to introduce diversity will be limited by the remaining available memory and storage spaces. To overcome these limitations, Mahmood and Shila [128] proposed a context-aware code partitioning and code diversity approach for IoT. IoT devices only store a minimal trusted code and depend on the context, another trusted code can be downloaded from a secure source (e.g. cloud), which can then be removed and replaced by other trusted code after this code is no longer needed. To further obfuscate the attacker's view, code diversification is used in conjunction with the context-aware. Different versions of the same code can be run at different times.

In the dynamic platform domain, while providing diversity through hardware diversity (e.g. dual processor architectures) may not be technically (i.e. in terms of the form factor of the sensors) nor economically feasible (i.e. in terms of cost), providing software diversity for the dynamic platform can be a viable option. Casola et al. [129] proposed an MTD approach based on fine-grained reconfiguration at two different architectural layers of WSN, namely the security layer and the physical layer. In the first reconfiguration scheme, the security layer protocol is dynamically changed between two or more cryptosystem implementations [130]. Similarly, the second reconfiguration scheme enables each node to swap to a new image that was stored on an external flash memory. In [130], the decision to change can be based on the rules such as reaching maximum execution time of application or non-reachability of a node.

24.5.2 MTD for IoT in the Dynamic Network Domain

Several MTD approaches that have been proposed for IoT in the recent years are mainly in the dynamic network domain [131–137]. In the following sections, a review of each work is organized into two categories, MTD based on Identity and non-Identity randomization. Figure 24.12 shows the proposed MTD for IoT classification.

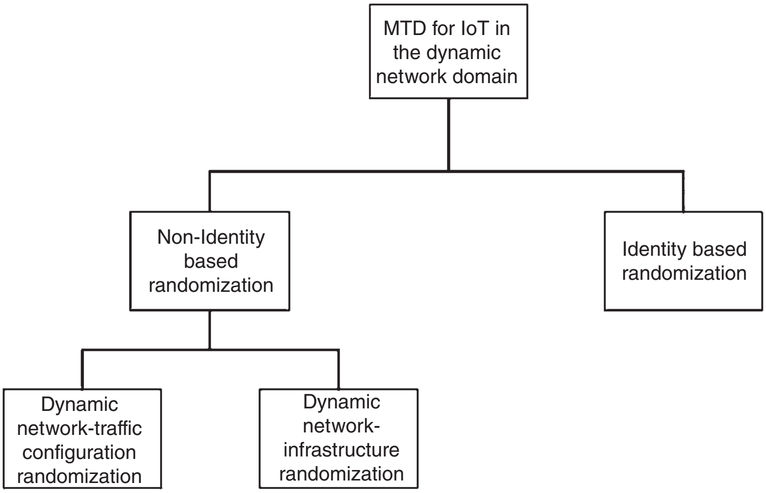

MTD for IoT in the dynamic network domain
Non-Identity based randomization
Identity based randomization
Dynamic network-traffic configuration randomization
Dynamic network-infrastructure randomization

Figure 24.12 Proposed MTD for IoT classification. The classification is derived from the existing works in the dynamic network domain specifically for IoT.

24.5.2.1 MTD Based on Identity Randomization

Depending on the IoT application's needs, MTD has started to be used in various IoT environments for more effective defense. Obviously, the explicit identities such as the physical and logical addresses of IoT devices become the main focus for randomization even though another identity may also exist [138]. With respect to the explicit addressing, network address shuffling is one of the first MTD methods comes to mind. It is basically changing IPv4 or IPv6 address of the devices in a network periodically. Thus, the attackers reach the wrong target or way better, a non-existent target. Judmayer et al. [135] assess the overhead and the effects of periodically changing network addresses and ports under various scenarios. In their experiments, they used Linux-based IoT systems such as Raspberry Pi (RasPi) [139], RasPi 2 [140], RasPi 3 [141], and Carambola 2 [142]. They measured the number of address change operations per second for each device. As expected, the more advanced one among these devices, which is RasPi3, outperformed the others. What is surprising is that Carambola 2 outperformed RasPi although RasPi has a clock rate of 700 MHz while Carambola 2 has a clock rate of 400 MHz. Also, they tested using multiple IP addresses simultaneously. They observed that it takes more time to add new addresses as the number of addresses already in use increases. Based on their findings, they concluded that it is feasible to shuffle network addresses in IoT environments.

24.5.2.1.1 MTD-Based IPv6 for IoT

Due to the huge amount of IoT devices, IPv6 is considered as the viable addressing scheme for IoT. However, a privacy issue may arise from the IPv6 stateless address autoconfiguration (SLAAC) feature that enables a host to self-assign a unique address. Typically, the 64 bits interface identifier (IID) portion of the IPv6 address is derived from the host MAC address in the form of 64-bit extended unique identifier (EUI-64) format. By utilizing this static IID, a third party can perform address tracking or traffic correlation. Additionally, with this static IID, targeted network attacks such as the address-based DoS and Man-in-the-middle (MITM) attacks become a security concern.

The Moving Target IPv6 Defense (MT6D) [102] is a non-deterministic IPv6 dynamic addressing that continually modifies the IP and port addresses of the sender and receiver of two communicating end-devices without breaking the existing connections. The aims are to preserve user privacy and protect against DoS and MITM attacks. Due to these features and the immense IPv6 address space, MT6D is a potential solution for Smart Grid [143] and for the low-powered, resource-constrained WSN running on IPv6 over Low-Powered Wireless Personal Area Network (6LoWPAN) [133, 134]. MT6D, however, is designed for full-scale systems and devices, and thus it needs to be adapted to be used for IoT constrained devices.

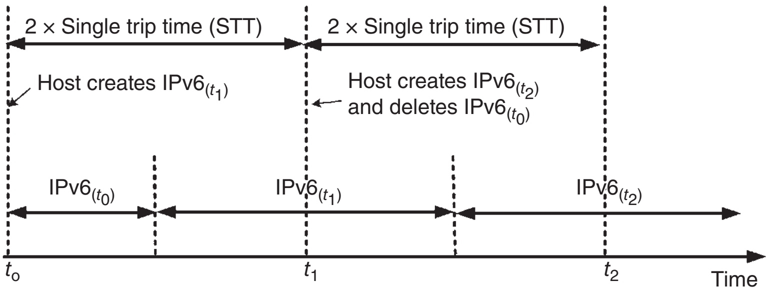
MT6D uses a hash function **H** to create a *vIID*$_x$, an obscured IID of a host *x*, from the first 64-bits of the hash value in Eq. (24.1). The input of **H** is the concatenation of (i) the real *IID*$_x$ for a host *x*, (ii) a shared symmetric key K_s, and (iii) the time t_i at instance *i*. The symmetric key is distributed between two communicating hosts through an out-of-band key exchanged. Besides for obscured IID generation, this key is also used for encryption.

$$vIID_{x(i)} = \mathbf{H}[IID_x \| K_s \| t_i]_{0\rightarrow 63} \tag{25.1}$$

$$vPort_{x(i)} = \mathbf{H}[IID_x \| K_s \| t_i]_{64\rightarrow 79} \tag{25.2}$$

The next 16-bits remaining unused bits of the hash value can be used as the obscured port number *vPort*$_x$ as in Eq. (24.2). The 128-bits MT6D IPv6 address is constructed by concatenating the global routing prefix (48 bits), subnet identifier (16 bits), and the obscured IID (64 bits). The minimum IID rotation time is at least twice of the single-trip time (STT) of a packet sent between a sender and receiver. Figure 24.13 illustrates the timeline of an obscured IPv6 address, from the creation to deletion.

When a new obscured IPv6 address is created, a host utilizes the Neighbor Discovery Protocol (NDP) to verify that there is no conflict with the existing addresses on the subnet and ensure that the routers have this new address in their routing table prior the usage of this address. This way, each host only stores two obscured IPv6 addresses, the current address and the next address, while routers maintain multiple obscured IPv6 addresses that refer to the same host. The deleted IPv6 addresses from hosts will then be removed from the router's routing table as well through the IPv6 internal mechanism. This new obscured IPv6 address, however, does not need to be disseminated to the other end of the communicating partner. Once the communicating partner knows the real IPv6 address of a host, it can



2 × Single trip time (STT)
2 × Single trip time (STT)
Host creates IPv6$_{(t_1)}$
Host creates IPv6$_{(t_2)}$ and deletes IPv6$_{(t_0)}$
IPv6$_{(t_0)}$
IPv6$_{(t_1)}$
IPv6$_{(t_2)}$
t_0
t_1
t_2
Time

Figure 24.13 The creation, usage, and deletion of an obscured IPv6 address timeline. A packet sent within one STT of the next rotation will use the obscured IPv6 address from the next rotation cycle to ensure that there is no additional overhead of connection reestablishment or breakdown.

calculate the obscured IPv6 address of its partner using the same hash function since it shares the same symmetric key. However, to find the correct obscured IPv6 address of the communicating partner, time synchronization between these two communicating devices is mandatory.

Instead of using the obscured IPv6 addresses to replace the address information in the original packet, MT6D creates a packet that consists of an MT6D header and the anonymized version of the original packet where both IP and MAC addresses of the source and destination are overwritten to make it anonymous and disable address tracking while keeping all other protocol information in place. The MT6D packet is sent from a source to a destination, either through un-encrypted or encrypted UDP tunnel. UDP is used rather than TCP for the tunneling to avoid any TCP connection establishment and termination each time MT6D address changes. The encrypted UDP tunnel prevents traffic correlation since the anonymized original packet is encrypted using the shared symmetric key and thus a third party cannot get any information from it. The MT6D's overhead is 62 bytes that come from 40 bytes of MT6D header, 8 bytes of UDP header, and 14 bytes of Ethernet frame header. MT6D can be implemented either as an embedded software on a host or as a stand-alone gateway device.

Several efforts to adapt MT6D for IoT has been recently performed [131–134]. These efforts strive to implement the dynamically changing IPv6 address scheme on the 6LoWPAN protocol. Three different locations (from the most constrained node to the least constrained node) have been identified as the candidate for the scheme operations [131]: (i) on the 6LoWPAN mote (i.e. sensor), (ii) the 6LoWPAN border router, and (iii) the IPv6 Gateway. However, only the first two locations are further studied [132, 133]. While 6LoWPAN mote is a resource-constrained device, the border router in a typical 6LoWPAN network is usually attached to an external power source and/or greater processing power, which enables it to connect to the Internet and acts as a bridge to the 6LoWPAN network. Thus, implementing the MT6D on the border router, on the one hand, offers a simpler design [132]. However, control of the border router is not always guaranteed since it can belong to a third party. Implementing the MT6D on a resource-constrained 6LoWPAN mote (i.e. the end-point), as MT6D was originally intended, ensures that an attacker cannot effectively capture the IPv6 traffic to the mote [132]. However, it requires some adjustments. For example, when MT6D is implemented on TMote Skye motes [144], a class-1 constrained node with 10 kB of RAM and 48 kB of ROM, using a full IPv6 stack is too expensive and thus, a lightweight version of the network stack (e.g. Rime stack from Contiki OS [145]) can be used [132].

The Micro-Moving Target IPv6 Defense (μ-MT6D) [133, 134] was designed to operate over 6LoWPAN with both modes of operation: a *host-based* mode and *border-based* mode. In the former, μ-MT6D is implemented at the end-point (i.e.

sensor), while in the latter, μ-MT6D is implemented at the border router. Initially, MT6D uses Secure Hash Algorithm 256 (SHA-256) that requires around 80 kB of storage. Thus, besides utilizing a lightweight operating system (e.g. Contiki OS), another effort that can be done to adapt MT6D for the resource-constrained node is by optimizing the SHA-256 or utilizing more lightweight cryptographic hash algorithms.

24.5.2.1.2 MTD-Based Identity Virtualization for MANET

Albanese et al. [138] proposed an MTD based Identity Virtualization mechanism for mobile ad-hoc networks (MANET), a persistent self-organizing infrastructure less network of mobile wireless nodes that allows each mobile node to join and leave the network at any time, with the aims of protecting the identity of mobile nodes and obstructing the reconnaissance phase from external attackers by using a dynamically changing virtual identity, instead of the mobile node real identity, for communications with other legitimate mobile nodes. This virtual identity is selected from a pool of virtual identities. A validity interval determines how long a virtual identity can be used before it must be replaced with a new virtual identity from the pool. This way, the virtual identity is dynamically changing, and the real identity of the mobile node is never publicly used.

Each mobile node generates a pool of N virtual identities using a hash chain, a method that can create N virtual identities from a single input value x by successively applying a cryptographic one-way hash function F to the input value N times. For example, $F^4(x) \equiv F(F(F(F(x))))$ represents a hash chain of length $N = 4$. To create a pool using the hash chain, each mobile node uses two input values for the hash function: (i) a mobile node generates a random initial seed value x_i, which is generated whenever a mobile node i needs to create a pool such as when a new pool is needed since the existing pool is exhausted; and (ii) a shared secret seed s, which is known by all legitimate mobile nodes. This shared secret seed is used to change the argument of the hash function at each recursive step using Eq. (24.3):

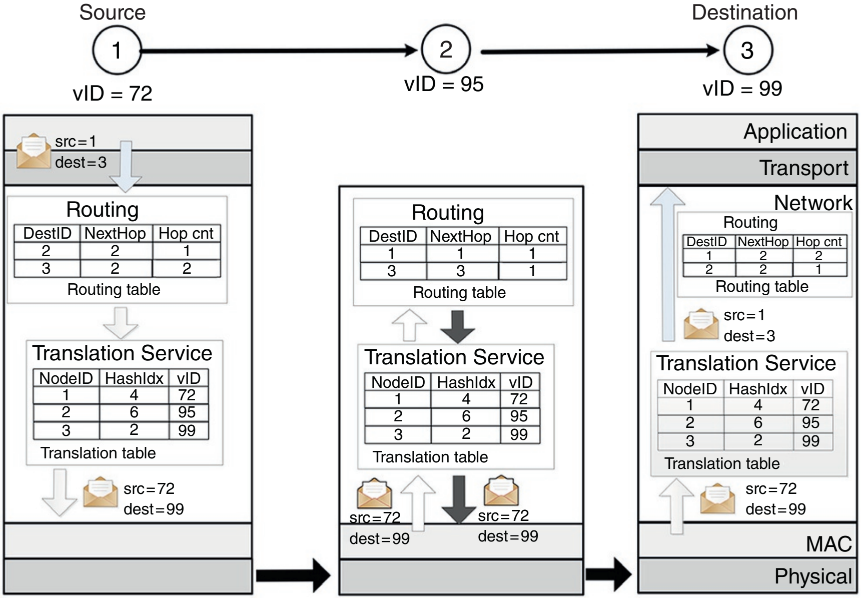
$$F^k(x_i) \equiv \begin{cases} x_i & \text{for } k = 0, \\ F\left(F^{(k-1)}\right)(x_i), s) & \forall k \in [1, N]. \end{cases} \tag{25.3}$$

The goal of using a shared secret seed is to prevent an attacker, who has the knowledge of the hash function, to perform a brute-force attack by using every initial seed value in the seed space to generate the entire hash chains. The virtual identity *vID* of a mobile node i is then selected from the hash chain in the reverse order of the hash chain generation (i.e. starting from the last element of the hash chain) as in Eq. (24.4):

$$vID_i(t) = F^{N-(t-1)}(x_i) \;\; \forall t \in [1, N]. \tag{25.4}$$

Thus, the first virtual identity will be $vID_i(1) = F^N(x_i)$, which is called the commitment of the chain, and the last virtual identity will be $vID_i(N) = F(x_i)$. For each $vID_i(t)$, a validity interval $T_i(t)$ is randomly selected from the validity range $[Tmin, Tmax]$.

To ensure that mobile nodes can communicate in this new dynamically changing identity environment without synchronization, the authors proposed to modify the Internet layer of TCP/IP protocol stack with the following mechanisms: (i) a translation service for identity mapping using a translation table; and (ii) an update protocol for the virtual identity dissemination and periodically updating the translation table. Each entry in the translation table stores the real identity of a legitimate mobile node, its current virtual identity, and the hash-index corresponding to the current virtual identity in the hash chain. When a sender node wants to send a message to a destination node, the translation service at the network layer translates the source and destination real identities in the message to their currently corresponding virtual identities before the message is broadcast. At any intermediate node located on the route to the destination, when this intermediate node receives this message, the translation service translates the virtual identities to their real identities to find the correct route to the destination. After the correct route is found, the message, which still using the virtual identities, is forwarded again to the next hop. Figure 24.14 illustrates the translation service operations at the network layer when a message is sent from the sender to the destination.



DestID	NextHop	Hop cnt
2	2	1
3	2	2

NodeID	HashIdx	vID
1	4	72
2	6	95
3	2	99

DestID	NextHop	Hop cnt
1	1	1
3	3	1

NodeID	HashIdx	vID
1	4	72
2	6	95
3	2	99

DestID	NextHop	Hop cnt
1	2	2
2	2	1

NodeID	HashIdx	vID
1	4	72
2	6	95
3	2	99

Figure 24.14 The translation service operations at the network layer.

When a mobile node *i* replaces its current virtual identity $vID_i(t)$ with a new $vID_i(t+1)$ since $T_i(t)$ expires, it sends a broadcast *Update* message that uses the new $vID_i(t+1)$ as the source identity and contains the hash index $N-(t-1)$ of that new identity in the mobile node *i*'s hash chain. On receiving the broadcast *Update* from *i*, each receiver must verify the authenticity of the sender *i* before the receiver can update its translation table. The verification is performed to the source virtual identity in the received *Update* message by checking that the equality in Eq. (24.5) holds.

$$F\left(F^{N-1}(x_i), s\right) \equiv F^{N}(x_i). \tag{25.5}$$

For example, when the sender *j* receives an *Update* message with a $vID(k)$ as the source identity and contains hash index *k* in the message, it will check for every entry *i* in its translation table whether the stored $HashIndex(i) > k$ and $F^{HashIndex(i)-k}(vID(k)) \equiv$ the currently stored *vID* in the translation table. If that is the case, node *i* is the originator of the *Update* message and the corresponding entry in the translation table is updated.

In addition, the authors also proposed the join and leave mechanisms since a mobile node can join or leave the network at any time. To join a network, a legitimate mobile node must have the shared secret *k* to encrypt its messages. The node first selects two values, a real identity *i* and an initial random seed x_i; generates a hash chain of *N* virtual identities; composes an encrypted *join request* message that contains the real identity *i* and a random number r_i in its payload and uses the commitment of the chain as the source address. The random number r_i is used by the node *i* to distinguish its own request from others in case two or more *join request* messages that are issued at the same time have either the same the real identity, the commitment of the chain, or both. On receiving a *join request* message, when a receiver recognizes that either of the real identity, commitment, or both is the same as its own, a *join response* message is broadcast to indicate a duplicate. Otherwise, this *join request* message is stored and marked as pending until a timer expires. When the timer expires and no *join response* message from other node related to that *join request* message is received, the receiver assumes the request is valid and stores the identity and commitment in its translation table. Otherwise, when a *join response* message is received, the *join request* message is discarded. For the sender, when a *join response* message is received for its *join request*, which indicates that either the identity or commitment is owned by another node, the sender must choose a new identity and secret key *s* and redo the join process.

When a node leaves the network, its network information (e.g. node identity, hash index, and current virtual identity) is still available and always valid in the translation table of other nodes in the network since there will be no more *update* message from the leaving node. For this case, a valid timer associated with each entry in the translation table is introduced. When this timer expires, which indicates that no more *update* messages are received within the valid time interval, any additional data packet using this identity is no longer considered as a legitimate packet.

24.5.2.1.3 MTD-Based DDoS Resistant Multicast (DRM)

Andrea et al. [137] modified the RPL, a routing protocol developed for low power and lossy networks and presented the Simple Agile RPL multiCAST (SARCAST) protocol against DDoS attacks. The protocol is based on an address agility technique called DDoS Resistant Multicast (DRM). Address agility technique prevents hostile actors from performing meaningful reconnaissance or an attack against a previously discovered target by shuffling destination address in any message. SARCAST benefits from the DRM concept and mitigates the hostile multicast traffic that targets the interiors of a deployed network. The agile addressing mechanism is embedded into the destination address field of the packet. Each system on the network periodically updates that field to the current valid address, and any packets with an expired or invalid address are rejected. The SARCAST was tested on a working IoT testbed, and results showed that the SARCAST can protect IoT systems against DDoS attacks with almost no adverse impact on overall performance and that the size of agile address history has a significant effect on packet delivery ratios.

24.5.2.2 MTD Based on Non-Identity Randomization

In this section, the MTD-based on the non-Identity randomization is presented and organized according to the three dynamic-network sub-classifications (if applicable): the dynamic path, dynamic network infrastructure, and dynamic network traffic configuration.

24.5.2.2.1 MTD Based Network Infrastructure Randomization

Tracking and localization services developed for IoT applications pose some risks to the location security of base-stations serving for WSNs in case of a compromised node in the network. This is because damaging a base-station may result in catastrophic consequences. To avoid such disasters, Chin and Xiong [136] proposed moving proximity base station defense (MPBSD) to conceal the location of a base station. MPBSD complicates localization techniques based on received signal-Strength-Indicator (RSSI). It presumes that multiple base-stations exist in the WSN, and one of them is elected as the active base station for a specific period. In the meantime, inactive base-stations transmit deceptive beacons to mask the location of the active base-station by thwarting localization methods. The real-world testbed experiments demonstrated that MPBSD is an effective MTD approach to provide obscurity in the location information of the base-station on duty in term of end-to-end delay.

24.5.2.2.2 MTD Based Network Traffic Configuration Randomization

Some recent study focused on Smart Grid domain where IoT-based smart meters with wireless transmission capabilities are deployed. Advanced Metering Infrastructure (AMI) is one of the Smart Grid applications that utilizes a smart meter to enable two-way communications between the household and utility company. AMI has enabled the utility company to perform its operations remotely such as energy consumption data collection, outage detection, and diagnostics. Due to its critical role and its predictable and deterministic behavior (e.g. periodic data collection schedule, same size data collection, and same route to convey data from the smart meter to the AMI headend systems), AMI has been the target of various attacks, called mimicry attacks. These types of attacks mimic the AMI behavior and follow the protocol. Due to the resource-constrained devices in AMI networks, deep packet inspection may not be possible and thus, these attacks can go undetected. Ali et al. [109], studied how MTD can be an effective proactive defense against the attacks by randomizing three configuration parameters of AMI: (i) report size, (ii) report interval, and (iii) relaying nodes.

Algin et al. [110] studied how MTD can be an effective means to eliminate selective jamming attacks for Smart Grid AMI networks. In this setup, jamming could be an important attack to block a smart meter's transmission if the schedule of the transmission is learned. To mitigate this issue, the authors proposed randomizing the schedules in each round of data transmissions so that the attackers will not be able to determine the exact transmission times. The randomization of the transmissions follows the idea of MTD, basically changing the times randomly for each smart meter. To this end, they employ the Fisher–Yates shuffle algorithm that has been shown to provide secure randomness. This algorithm guarantees that a smart meter will transmit in a different time slot than its previous slot. The experiment results indicate that the proposed MTD approach does not bring any significant overhead. On the contrary, the authors demonstrate that in some cases MTD helps to improve packet deliver ratios as the randomness help in accessing the wireless channel more efficiently. In the same lines, there was another very recent study which explored MTD for vehicular environments [146]. The idea is transmitting data across dynamic multi paths relayed through vehicles traveling on a multi-lane road. Again, the study demonstrated that jamming a targeted data stream would be very difficult.

24.5.3 SDN and MTD for IoT

SDN architecture has been considered a cost-effective solution that can ease the network management and dynamic configuration for the IoT devices that are in many cases not easily accessible. Once SDN is employed, it can pave the way to also apply MTD techniques for IoT applications. In this subsection, we first explore the research efforts which strive to apply SDN to IoT environments and then make a case for SDN-based MTD for certain IoT applications.

24.5.3.1 SDN for IoT Environments

Besides for cyber resilience in IIoT and cybersecurity deception in IoT as presented in Chapters 20 and 21 respectively, there are some recent works on possible use-cases, architecture, implementation, and even security of SDN integration for IoT environments.

For instance, the authors in [147] discuss two use-cases of SDN in IoT environments. In the first one, they suggest using OpenFlow-enabled switches in gateways for smart homes and claim that it will decrease the cost of the management and maintenance of the network. Their proposal framework is shown in Figure 24.15.

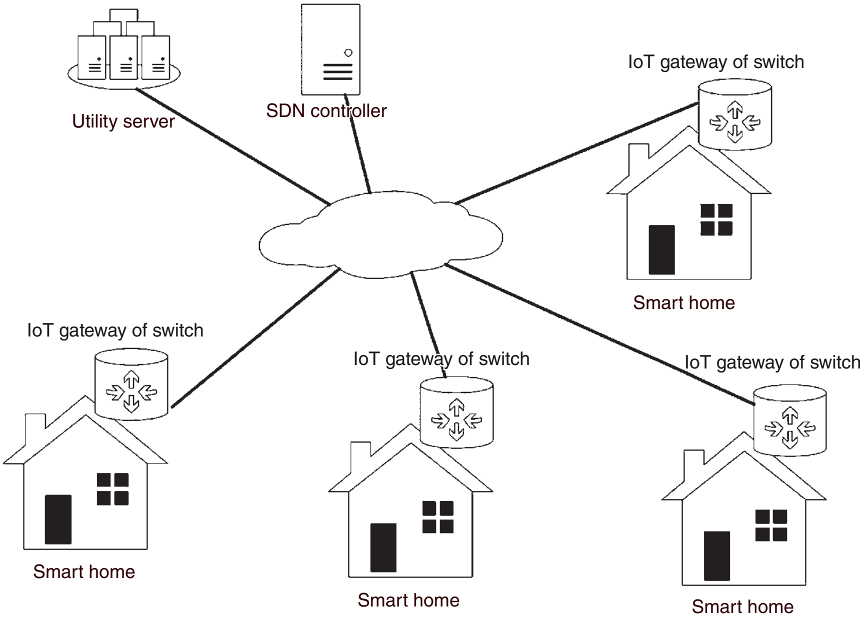


Figure 24.15 SDN-based switches for Smart Home.

As a second use-case, the authors consider SDN-enabled Evolved Packet Core (EPC) in LTE networks in order to separate and handle efficiently the traffic that must pass through EPC (e.g. voice traffic), and traffic that can be offloaded to the Internet (e.g. video traffic). This simple solution reduces the load on EPC and the operating expenditures for the service provider. While the second example relates mostly to EPC, the first use-case is a perfect fit solely for IoT environments. In another study [148], researchers demonstrate the scenarios and standards for virtualization and SDN in IoT environments. They argue that SDN would provide a flexible and interoperable environment which is critical for IoT communications. They show SDN on Wireless Mesh Network (WMN) by connecting SDN Controller to Wireless Mesh Routers (WMRs). In addition, they also mention that virtualization cannot be directly applied to wireless environments due to the dynamic nature of air interface.

There are a couple of architectural frameworks for SDN-based IoT that researchers have proposed. For instance, Zhijing et al. proposed a layered controller design [149]. He showed that their proposed generic algorithm for flow scheduling has better performance in terms of throughput, delay, and jitter compared to two common scheduling algorithms, namely bin packing and load balancing. Alejandro et al. introduced Software Defined Wireless Sensor Network (SDWSN) where SDN Controller is considered in the base station [150]. They follow the flow table idea of OpenFlow and try to solve the compatibility issue of other SDN nodes. Different studied also proposed to change OpenFlow to support Wireless Sensor Networks (WSNs) [151, 152]. The approaches in these works can control the flow of data. However, SDN Controller could also be used for controlling and managing the sensor hardware as it would change sensors' status depending on the need. Meanwhile, the authors in [153] came up with a framework for integrating SDN and Fog Computing in the IoT domain. Smart transportation, video surveillance, and precision agriculture are presented as use-cases of such framework. There is also an effort on providing communication between IoT devices and SDN Controller. They propose REST protocol to turn IoT devices to Web resources [154]. REST interface is used as Northbound API from SDN Controller.

Another noteworthy IoT application domain for SDN deployment is vehicular networks. SDN has started to be applied to Vehicular environments in recent years [155]. Among these efforts, there were a few works which focused on routing, bandwidth management, QoS, etc. among vehicles which are considered as IoT devices. As an earlier work in this subject, authors in [156] introduced SDN-based communication for vehicular devices. They propose LTE-based connection for control plane and Wi-Fi for data plane. Their solution provides better packet delivery ratio compared to Adhoc on Demand Distance Vector Routing (AODV) [157], On-demand Link State Routing (OLSR) [158], and Destination Sequence Distance Vector (DSDV) [159] routing protocols considering quick response mechanism of

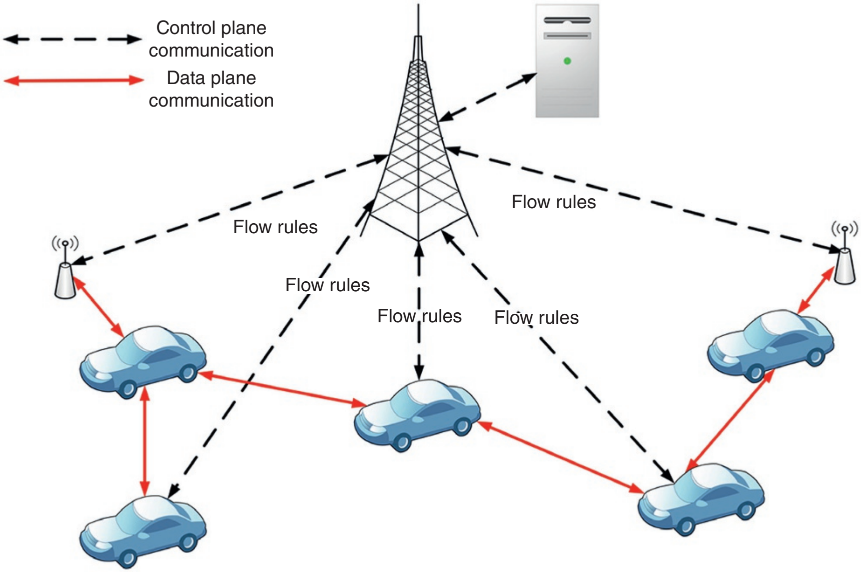


Figure 24.16 SDN-based vehicular network.

SDN Controller in case of topology changes. As a different work in this area, Baihong et al. [160] introduced SDN-based on-demand routing protocol (SDOV) with two level design (two-level controller); one for deciding which path to forward by utilizing vehicle information, and the other one to select forwarding vehicles according to the decision made in the upper layer (see Figure 24.16). Their simulation demonstrates better results on the impact of vehicle density, speed on data transmission rate and average packet delay compared to common ad-hoc routing protocols namely OLSR, Dynamic Source Routing (DSR) [161], DSDV, etc.

24.5.3.2 SDN-Based MTD for IoT

As was shown in the previous studies, applying SDN and MTD at the same time to IoT environments may not always be a feasible operation due to resource constraints and operational needs of the applications. In this section, we advocate two specific applications of IoT where SDN-based MTD can be an effective means for secure and dependable operations.

24.5.3.2.1 Internet of Battlefield Things

One of such applications is the Internet of Battlefield Things (IoBT) which focuses on the military applications where various IoT devices (e.g. sensors, munitions, weapons, vehicles, robots, and human-wearable devices) and human warfighters need to communicate in real-time through wireless communications [162]. Mobility is the inherent nature of this ad hoc network. The command and control messages flow through the network either in a multi-hop manner or single-hop depending on the priorities and needs. As critical information is shared within this network, securing the communication infrastructure is essential. While sophisticated security approaches and tools can be employed, the resource-constrained "things" (IoT devices) do not allow for such solutions. In addition, fault-tolerance or availability of IoT devices are also very critical due to the adverse nature of the battlefield environments. While these devices will have the capability to act autonomously with some AI features, centralized management capabilities are also important for more effective defense and strategy spreading. In this vein, SDN and MTD can be a perfect fit with their centralized nature which would also in-line with the hierarchical aspects of military infrastructure and information.

24.5.3.2.2 Industrial Internet of Things

The other potential application is the IIoT environments. IIoT systems are perfect examples of CPS and they are often identified as CIs for their national importance to national economy and security. Security of these IIoT systems is essential, especially in the wake of cyber warfare, including Advanced Persistent Threats (APT) and nation-state attacks. MTD, in addition to the static security measures, can provide an edge to fight against incessant and highly-capable state-backed cybercriminals, who mostly exploits zero-day vulnerabilities. IIoT systems are often equipped with communication infrastructures for control channels, which easily allow the deployment of SDN to employ MTD. However, these communication systems are often large, widespread, with legacy devices and delicate industrial protocols. Hence, the deployment of SDN, considering technical difficulties, limited resources, MTD measures, and security requirements, is challenging, creating an interesting decision-making problem.

In an IIoT environment, the cyberinfrastructure is highly integrated with physical systems through control routines. MTD traditionally considers random and/or periodic, often proactive, moves of cyber components, protocols, or behaviors. The physical moves, if feasible, can open a new dimension to MTD. A successful attack against an IIoT system often needs the knowledge of the physical properties, physical agility can thwart such attacks. A pioneer work of this kind has been proposed in a smart grid scenario [163]. An integration of physical agility with cyber moves will provide a powerful MTD technique (Figure 24.17a). The design of the MTD technique (particularly, the selection and timing of MTD moves/actions) for an IIoT system must consider the IIoT control structure. An MTD move can easily deteriorate the system's operation. While the impact of a move needs to be within the optimal operational requirement, the move must affect the adversary's capability. In the case of the integration of cyber and physical MTD moves, both types

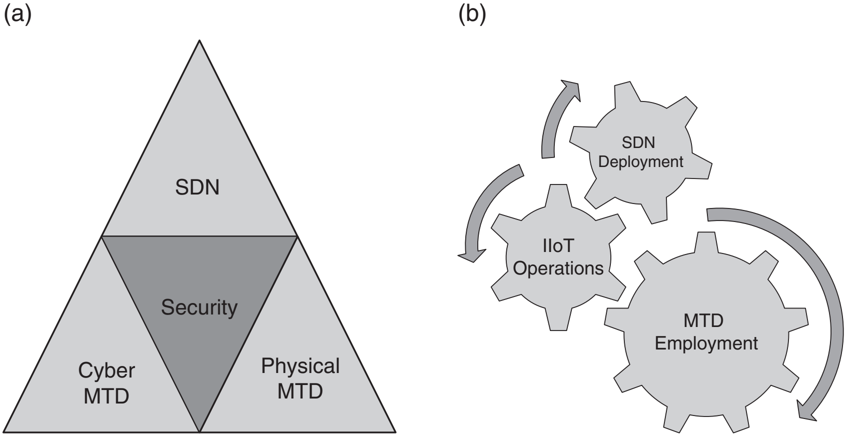


Figure 24.17 (a) MTD for IIoT systems and (b) optimal design of an MTD technique.

of actions must comply with each other such that one kind should not negatively influence the other, rather collectively offer a larger impact. An SDN-based control structure can be leveraged to govern these MTD actions for an effective impact. The optimal design of an MTD technique, as shown in Figure 24.17b, needs to consider the MTD moves, IIoT operational requirements, and SDN deployment factors.

24.6 Future Research Challenges

A full application of MTD techniques for IoT specifically for IoBT and IIoT comes with additional challenges that necessitate new research regardless of the use of SDN or not. In addition, any other IoT application might benefit from MTD while still there are challenges regarding the availability of resources. In this section, we elaborate on these research challenges as listed below:

- *SDN Control Channel in IoBT:* IoBT requires constant communication among a possibly large number of mobile nodes. Any attack on this network will thus affect almost all the nodes in terms of information gathering, command and control, and resources. Therefore, a fast access to these devices from the SDN controller is needed. Given that this channel needs to be wireless, reliability and availability of the communications will be a challenge. Security and robustness would be a must for this communication. Therefore, new wide area control protocols are needed. The emerging 5G technologies have the potential to address this challenge with its reliability and long-distance coverage. However, 5G may not be available everywhere and thus its features such as D2D as well as technologies from 802.11 families such as IEEE 802.11ah need to be investigated in terms of QoS they can guarantee.

- *Local Intelligence in IoBT:* Given the challenges with the control channel reliability, the nodes should be given some local intelligence so that they can operate when SDN controller is not accessible. The level of intelligence should be subject to application requirements. This intelligence can be dynamic and based on machine learning technologies where the devices can learn from their environments to act without instructions.
- *MTD-Aware SDN Deployment in IIoT:* The deployment of SDN in an existing network (e.g. replacing the traditional routers with SDN-enabled switches) is often restrained by limited resources, legacy systems, and/or technical constraints. Therefore, the challenge of deploying (often incremental) the SDN architecture within the limits while optimally achieving the security/defense objective should be explored under different parameters.
- *Integrated Cyber and Physical MTD Measures in IIoT:* While the cyber and physical moves together can bring a larger MTD capability, these actions must comply with one another, without declining the optimal operation of the IIoT system. Therefore, an important research direction for IIoT security is to explore the feasible physical moves and their MTD effectiveness, as well as to study the optimal integration of physical agility measures with the traditional cyber moves in the SDN-based MTD framework.
- *MTD Games:* Given the resource constraints and operational requirements, MTD in IoBT/IIoT may not be able to arbitrate many properties extensively. Therefore, game-theoretical approaches are needed to deceive the attackers based on their communication and action patterns. This will introduce a trade-off between the resources, service constraints, technical/communication feasibility, and the attacker's assumed capabilities. Another interesting piece of this challenge is to be able to model the behavior of the attackers in certain conditions so that MTD moves could be better arranged.
- *Smart Moving Target Defense for IoT:* Most proposed MTD approaches are typically not an attacker-aware and the decision to select the move is based on the pre-defined constraints. IoT, on the other hand, produces a huge amount of data, which is known as the big data. The IoT data analytics is much powerful than the traditional data analytics since the IoT data analytics are intended to do the real-time processing before the data becomes irrelevant or obsolete. Typically, the IoT data analytics can be placed at the edge (i.e. fog/edge computing) or at the cloud. Therefore, the IoT data analytics can be exploited to support a smart attacker-aware MTD as opposed to traditional networks.
- *Collaborative Multi-Attribute Moving Target Defense:* In this chapter, we have shown that most MTD approaches concentrate only on a single attribute (e.g.

address, path, configuration, etc.). Only a few approaches attempt to utilize more than one attribute for the proactive defense. Therefore, a collaborative multi-attribute MTD can be a potential future research direction.

24.7 Conclusion

In this chapter, we provided a survey of existing MTD mechanisms geared for IoT environments. We first provided an overview of MTD and present a classification of MTD mechanisms in general before we make the transition to IoT environments. With respect to IoT, we first discussed how MTD could be fit to IoT for security and defense purposes. We demonstrated that despite the overview of MTD, there are many applications of MTD for IoT security that may provide defense against a diverse set of attacks including zero-days. Our specific focus on the topic was IoBT and IIoT which we claim to welcome SDN-based MTD for a comprehensive security that will be flexible and cost-effective. We laid off several future research challenges that will be of value to new researchers and industry. Overall, we advocated that for IoBT and IIoT, MTD can be an effective defense when it is integrated with SDN.

References

1 Rajkumar Buyya and Amir Vahid Dastjerdi. *Internet of Things: Principles and Paradigms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2016.

2 J. A. Guerrero-ibanez, S. Zeadally, and J. Contreras-Castillo. Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and internet of things technologies. *IEEE Wireless Communications*, 22(6):122–128, Dec 2015.

3 N. Bui, A. P. Castellani, P. Casari, and M. Zorzi. The internet of energy: a web-enabled smart grid system. *IEEE Network*, 26(4):39–45, July 2012.

4 N. Suri, M. Tortonesi, J. Michaelis, P. Budulas, G. Benincasa, S. Russell, C. Stefanelli, and R. Winkler. Analyzing the applicability of internet of things to the battlefield environment. In *2016 International Conference on Military Communications and Information Systems (ICMCIS)*, pages 1–8, May 2016.

5 D. Singh, G. Tripathi, A. M. Alberti, and A. Jara. Semantic edge computingand IoT architecture for military health services in battlefield. In *2017 14th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 185–190, Jan 2017.

6 F. T. Johnsen, Z. Zieliski, K. Wrona, N. Suri, C. Fuchs, M. Pradhan, J. Furtak, B. Vasilache, V. Pellegrini, M. Dyk, M. Marks, and M. Krzyszto. Application of IoT in military operations in a smart city. In *2018 International Conference on Military Communications and Information Systems (ICMCIS)*, pages 1–8, May 2018.

7 Jonathan Lampe. *IoT Security: Locking Down the Internet of Things*. Auerbach Publications, Boston, MA, USA, 1st edition, 2017.

8 H. Okhravi, T. Hobson, D. Bigelow, and W. Streilein. Finding focus in the blur of moving-target techniques. *IEEE Security Privacy*, 12(2):16–26, Mar 2014.

9 Nick Feamster, Jennifer Rexford, and Ellen Zegura. The road to sdn: An intellectual history of programmable networks. *ACM SIGCOMM Computer Communication Review*, 44(2):87–98, 2014.

10 Kevin Ashton et al. That internet of things thing. *RFID Journal*, 22 (7):97–114, 2009.

11 D INFSO. Networked enterprise & rfid infso g. 2 micro & nanosystems. In *Internet of Things in Cooperation with the Working Group RFID of the ETP EPOSS,* 2020, 4.

12 Linus Wallgren, Shahid Raza, and Thiemo Voigt. Routing attacks and countermeasures in the rpl-based internet of things. *International Journal of Distributed Sensor Networks*, 9(8):794326, 2013.

13 Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.

14 Lane Thames and Dirk Schaefer. Industry 4.0: An overview of key benefits, technologies, and challenges. In *Cybersecurity for Industry 4.0*, pages 1–33. Springer, 2017.

15 Arijit Karati, SK Hafizul Islam, and Marimuthu Karuppiah. Provably secure and lightweight certificateless signature scheme for IIoT environments. *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pages 3701–3711, Aug. 2018.

16 Rwan Mahmoud, Tasneem Yousuf, Fadi Aloul, and Imran Zualkernan. Internet of things (IoT) security: Current status, challenges and prospective measures. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST),* pages 336–341. IEEE, 2015.

17 Shahid Mumtaz, Ahmed Alsohaily, Zhibo Pang, Ammar Rayes, Kim Fung Tsang, and Jonathan Rodriguez. Massive internet of things for industrial applications: Addressing wireless IIoT connectivity challenges and ecosystem fragmentation. *IEEE Industrial Electronics Magazine*, 11(1):28–33, 2017.

18 Dhananjay Singh, Gaurav Tripathi, and Antonio J. Jara. A survey of internet-of things: Future vision, architecture, challenges and services. In *2014 IEEE world forum on Internet of things (WF-IoT),* pages 287–292. IEEE, 2014.

19 Arsalan Mosenia and Niraj K. Jha. A comprehensive study of security of internet of-things. *IEEE Transactions on Emerging Topics in Computing*, 5 (4):586–602, 2017.

20 Dave Evans. The internet of things: How the next evolution of the internet is changing everything, 2011. URL: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

21 Shuo Feng, Peyman Setoodeh, and Simon Haykin. Smart home: Cognitive interactive people-centric internet of things. *IEEE Communications Magazine*, 55 (2):34–39, 2017.

22 Samet Tonyali, Ozan Cakmak, Kemal Akkaya, Mohamed MEA Mahmoud, and Ismail Guvenc. Secure data obfuscation scheme to enable privacy-preserving state estimation in smart grid AMI networks. *IEEE Internet of Things Journal*, 3(5):709–719, 2016.

23 Hawzhin Mohammed, Samet Tonyali, Khaled Rabieh, Mohamed Mahmoud, and Kemal Akkaya. Efficient privacy-preserving data collection scheme for smart grid ami networks. In *2016 IEEE Global Communications Conference (GLOBECOM),* pages 1–6. IEEE, 2016.

24 Samet Tonyali, Kemal Akkaya, Nico Saputro, A. Selcuk Uluagac, and Mehrdad Nojoumian. Privacy-preserving protocols for secure and reliable data aggregation in IoT-enabled smart metering systems. *Future Generation Computer Systems*, 78:547–557, 2018.

25 Samet Tonyali. *Privacy-Preserving Protocols for IEEE 802.11s-based Smart Grid Advanced Metering Infrastructure Networks*. PhD thesis, Florida International University, 2018.

26 Ahmad Alsharif, Mahmoud Nabil, Samet Tonyali, Hawzhin Mohammed, Mohamed Mahmoud, and Kemal Akkaya. Epic: Efficient privacy-preserving scheme with e2e data integrity and authenticity for ami networks. *arXiv preprint arXiv:1810.01851*, 2018.

27 Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. Security and privacy challenges in industrial internet of things. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC),* pages 1–6. IEEE, 2015.

28 Amin Hassanzadeh, Shimon Modi, and Shaan Mulchandani. Towards effective security control assignment in the industrial internet of things. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT),* pages 795–800. IEEE, 2015.

29 Michele Albano, Jos´e Bruno Silva, and Luis Lino Ferreira. The industrial internet of things. *22^o Seminário da Rede Temática de Comunicações Móveis*, 2017.

30 Priya. IoT building blocks and architecture: IoT part 2, 2012. URL: https://www.engineersgarage.com/Articles/Internet-of-Things-Architecture

31 ZigBee Alliance. IEEE 802.15. 4, Zigbee standard, 2009.

32 Z-Wave Alliance. About z-wave technology. Z-Wave Alliance, Technical Report, 2013.

33 Yasir Zaki. Long term evolution (lte). In *Future Mobile Communications*, pages 13–33. Springer, 2013.

34 Behrouz A. Forouzan and Sophia Chung Fegan. *TCP/IP Protocol Suite*. McGrawHill Higher Education, 2002.

35 Statista. Internet of things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions), 2018. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

36 Arunabha Ghosh, David R. Wolter, Jeffrey G. Andrews, and Runhua Chen. Broadband wireless access with wimax/802.16: Current performance benchmarks and future potential. *IEEE Communications Magazine*, 43(2):129–136, 2005.

37 Brian P. Crow, Indra Widjaja, Jeong Geun Kim, and Prescott T. Sakai. IEEE 802.11 wireless local area networks. *IEEE Communications Magazine*, 35(9):116–126, 1997.

38 SIG Bluetooth. Bluetooth specification, 2003.

39 SANS Institute InfoSec Reading Room. *The 2018 Sans Industrial IoT Security Survey: Shaping IIoT Security Concerns*. SANS Institute, 2018.

40 Jon Postel et al. Rfc 791: Internet protocol, 1981.

41 RIPE NCC. Understanding ip addressing and cidr charts, 2016.

42 Steve Deering and Robert Hinden. Internet protocol, version 6 (ipv6) specification. Technical report, 2017.

43 Geoff Mulligan. The 6lowpan architecture. In *Proceedings of the 4th Workshop on Embedded Networked Sensors*, pages 78–82. ACM, 2007.

44 Jose A. Gutierrez, Marco Naeve, Ed Callaway, Monique Bourgeois, Vinay Mitter, and Bob Heile. IEEE 802.15. 4: A developing standard for low-power low-cost wireless personal area networks. *IEEE Network*, 15(5):12–19, 2001.

45 Gabriel Montenegro, Nandakishore Kushalnagar, Jonathan Hui, and David Culler. Transmission of ipv6 packets over IEEE 802.15. 4 networks. Technical report, 2007.

46 Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. Technical report, 1999.

47 Eric Rescorla. Http over tls. Technical report, 2000.

48 Jon Postel. Transmission control protocol. Technical report, 1981.

49 Zach Shelby, Klaus Hartke, and Carsten Bormann. The constrained application protocol (coap). Technical report, 2014.

50 Jon Postel. User datagram protocol. Technical report, 1980.

51 Roy Fielding. Representational state transfer. In *Architectural Styles and the Design of Network-based Software Architecture*, pages 76–85, 2000. https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf

52 Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. Mqtt-sa publish/subscribe protocol for wireless sensor networks. In *3rd International Conference on Communication Systems Software and Middleware and Workshops, 2008. comsware 2008,* pages 791–798. IEEE, 2008.

53 Apache Hive. Live long and process (llap), 2018. URL: https://gerardnico.com/db/hive/llap

54 Peter Saint-Andre. Extensible messaging and presence protocol (xmpp): Core. Technical report, 2011.

55 Steve Vinoski. Advanced message queuing protocol. *IEEE Internet Computing*, Vol. 10, no. 6, pp. 87–89, Nov.–Dec. 2006.

56 Eric Rescorla and Nagendra Modadugu. Datagram transport layer security version 1.2. Technical report, 2012.

57 Sean Turner. Transport layer security. *IEEE Internet Computing*, 18 (6):60–63, 2014.

58 T. Winter. Routing protocol for low-power and lossy networks. Technical report, rfc 6550, 6551, 6552. IETF, 2012.

59 Jose A. Gutierrez, Edgar H. Callaway, and Raymond L. Barrett. *Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensors with IEEE 802.15. 4*. IEEE Standards Association, 2004.

60 Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of Bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753, 2012.

61 Regis J. Bates. *GPRS: General Packet Radio Service*. McGraw-Hill Professional, 2001.

62 Ahmad Jalali, Witold Krzymien, and Paul Mermelstein. Medium access control scheme for data transmission on code division multiple access (cdma) wireless systems, Oct 27, 1998. US Patent 5,828,662.

63 Simon S. Lam. A carrier sense multiple access protocol for local networks. *Computer Networks*, 4(1):21–32, 1980.

64 Fouad A. Tobagi and V. Bruce Hunt. Performance analysis of carrier sense multiple access with collision detection. *Computer Networks (1976)*, 4 (5):245–259, 1980.

65 Souvik Sen, Romit Roy Choudhury, and Srihari Nelakuditi. Csma/cn: Carrier sense multiple access with collision notification. *IEEE/ACM Transactions on Networking (ToN)*, 20(2):544–556, 2012.

66 Roy Want. Near field communication. *IEEE Pervasive Computing*, 10 (3), pp. 4–7, July–September 2011.

67 Ivan Muller, Joao Cesar Netto, and Carlos Eduardo Pereira. Wirelesshart field devices. *IEEE Instrumentation & Measurement Magazine*, 14 (6), pp. 20–25, December 2011.

68 Juan Carlos Zuniga and Benoit Ponsard. Sigfox system description. LPWAN@ IETF97, Nov. 14, 2016.

69 Maarten Weyn, Glenn Ergeerts, Rafael Berkvens, Bartosz Wojciechowski, and Yordan Tabakov. Dash7 alliance protocol 1.0: Low-power, mid-range sensor and actuator communication. In *2015 IEEE Conference on Standards for Communications and Networking (CSCN),* pages 54–59. IEEE, 2015.

70 Ferran Adelantado, Xavier Vilajosana, Pere Tuset-Peiro, Borja Martinez, Joan Melia-Segui, and Thomas Watteyne. Understanding the limits of lorawan. *IEEE Communications Magazine*, 55(9):34–40, 2017.

71 Wikipedia. Thread (network protocol), 2018.

72 Paul Darbee. Insteon the details, smarthouse. Inc., Aug, 11:68, 2005.

73 Matthias Kovatsch, Simon Duquennoy, and Adam Dunkels. Erbium (er) rest engine and coap implementation for contiki, 2014.

74 Olaf Bergmann. Tinydtls. https://projects.eclipse.org/projects/iot.tinydtls, pages 2–15, 2013.

75 D. Gothberg. Micro-ip for embedded systems. *Computer Club West*, 2005. https://tools.ietf.org/html/draft-gothberg-micro-ip-00

76 Naganand Doraswamy and Dan Harkins. *IPSec: The New Security Standard for the Internet, Intranets, and Virtual Private Networks*. Prentice Hall Professional, 2003.

77 Nicolas Tsiftes, Joakim Eriksson, and Adam Dunkels. Low-power wireless ipv6 routing with Contiki rpl. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 406–407. ACM, 2010.

78 A. Dunkels. Sicslowpan-internet-connectivity for low-power radio systems. *SICS*, 2008. https://internetstiftelsen.se/docs/SICS_Lowpan-report.pdf

79 Adam Dunkels. The Contiki mac radio duty cycling protocol, 2011. URL: http://dunkels.com/adam/dunkels11contikimac.pdf

80 Muhammad Omer Farooq and Thomas Kunz. Contiki-based IEEE 802.15. 4 node's throughput and wireless channel utilization analysis. In *Wireless Days (WD), 2012 IFIP*, pages 1–3. IEEE, 2012.

81 Joachim Feld. Profinet-scalable factory communication for all applications. In *2004 IEEE International Workshop on Factory Communication Systems, 2004. Proceedings*, pages 33–38. IEEE, 2004.

82 IDA Modbus. Modbus application protocol specification v1. 1a. *North Grafton, Massachusetts*. URL: www.modbus.org/specs.php, 2004.

83 Thomas Ulz, Thomas Pieber, Christian Steger, Sarah Haas, Holger Bock, and Rainer Matischek. Bring your own key for the industrial internet of things. In *2017 IEEE International Conference on Industrial Technology (ICIT),* pages 1430–1435. IEEE, 2017.

84 Gerardo Pardo-Castellote. Omg data-distribution service: Architectural overview. In *Proceedings. 23rd International Conference on Distributed Computing Systems Workshops, 2003*, pages 200–206. IEEE, 2003.

85 John Matherly. Shodan search engine. URL: https://www.shodan.io, 2009.

86 Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. DDoS in the IoT: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.

87 Kieren McCarthy. California cracks down on internet of crap passwords with new law to stop the botnets, 2018. https://www.theregister.co.uk/2018/10/04/california_iot_password/

88 Chad Perrin. The cia triad, 2008. URL: https://www.techrepublic.com/blog/it-security/the-cia-triad/

89 Christian Lesjak, Holger Bock, Daniel Hein, and Martin Maritsch. Hardware secured and transparent multi-stakeholder data exchange for industrial IoT. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN),* pages 706–713. IEEE, 2016.

90 Christian Lesjak, Daniel Hein, and Johannes Winter. Hardware-security technologies for industrial IoT: Trustzone and security controller. In *Industrial Electronics Society, IECON 2015-41st Annual Conference of the IEEE*, pages 002589–002595. IEEE, 2015.

91 Sreejaya Viswanathan, Rui Tan, and David KY Yau. Exploiting power grid for accurate and secure clock synchronization in industrial IoT. In *2016 IEEE Real-Time Systems Symposium (RTSS),* pages 146–156. IEEE, 2016.

92 Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: Enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

93 Floodlight Project. Floodlight controller, 2014. URL: http://www.projectfloodlight.org/floodlight/

94 Opendaylight. Opendaylight, 2018. URL: https://www.opendaylight.org/

95 Mark Masse. *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces*. O'Reilly Media, Inc., 2011.

96 NITRD. National cyber leap year summit 2009, co-chairs report. Technical Report, Federal Networking and Information Technology Research and Development (NITRD) Program, 2009.

97 Yue-Bin Luo, Bao-Sheng Wang, Xiao-Feng Wang, Xiao-Feng Hu, and Gui-Lin Cai. TPAH: A universal and multi-platform deployable port and address hopping mechanism. In *2015 International Conference on Information and Communications Technologies (ICT 2015)*, pages 1–6, April 2015.

98 Y. B. Luo, B. S. Wang, X. F. Wang, X. F. Hu, G. L. Cai, and H. Sun. RPAH: Random port and address hopping for thwarting internal and external adversaries. In *Trustcom/BigDataSE/ISPA, 2015 IEEE*, volume 1, pages 263–270, Aug 2015.

99 Ehab Al-Shaer, Qi Duan, and Jafar Haadi Jafarian. *Random Host Mutation for Moving Target Defense*, pages 310–327. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

100 Jafar Haadi H. Jafarian, Ehab Al-Shaer, and Qi Duan. Spatio-temporal address mutation for proactive cyber agility against sophisticated attackers. In *Proceedings of the First ACM Workshop on Moving Target Defense*, MTD '14, pages 69–78, New York, NY, USA, 2014. ACM.

101 J. H. Jafarian, E. Al-Shaer, and Q. Duan. Adversary-aware ip address randomization for proactive agility against sophisticated attackers. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 738–746, April 2015.

102 M. Dunlop, S. Groat, W. Urbanski, R. Marchany, and J. Tront. The blind man's bluff approach to security using IPv6. *IEEE Security Privacy*, 10(4):35–43, July 2012.

103 S. Yan, X. Huang, M. Ma, P. Zhang, and Y. Ma. A novel efficient address mutation scheme for ipv6 networks. *IEEE Access*, 5:7724–7736, 2017.

104 Qi Duan, Ehab Al-Shaer, and Haadi Jafarian. Efficient random route mutation considering flow and network constraints. In *2013 IEEE Conference on Communications and Network Security (CNS),* pages 260–268. IEEE, 2013.

105 Usman Rauf, Fida Gillani, Ehab Al-Shaer, Mahantesh Halappanavar, Samrat Chatterjee, and Christopher Oehmen. Formal approach for resilient reachability based on end-system route agility. In *Proceedings of the 2016 ACM Workshop on Moving Target Defense*, MTD'16, pages 117–127, New York, NY, USA, 2016. ACM.

106 Huangxin Wang, Quan Jia, Dan Fleck, Walter Powell, Fei Li, and Angelos Stavrou. A moving target ddos defense mechanism. *Computer Communications*, 46:10–21, 2014.

107 P. Wood, C. Gutierrez, and S. Bagchi. Denial of service elusion (dose): Keeping clients connected for less. In *2015 IEEE 34th Symposium on Reliable Distributed Systems (SRDS)*, pages 94–103, Sept 2015.

108 S. Venkatesan, M. Albanese, K. Amin, S. Jajodia, and M. Wright. A moving target defense approach to mitigate ddos attacks against proxy-based architectures. In *2016 IEEE Conference on Communications and Network Security (CNS)*, pages 198–206, Oct 2016.

109 M. Q. Ali, E. Al-Shaer, and Q. Duan. Randomizing AMI configuration for proactive defense in smart grid. In *2013 IEEE International Conference on Smart Grid Communications (SmartGridComm),* pages 618–623, Oct 2013.

110 Ramazan Algin, Huseyin O. Tan, and Kemal Akkaya. Mitigating selective jamming attacks in smart meter data collection using moving target defense. In *Proceedings of the 13th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, Q2SWinet'17, pages 1–8, New York, NY, USA, 2017. ACM.

111 Jafar Haadi Jafarian, Ehab Al-Shaer, and Qi Duan. Openflow random host mutation: Transparent moving target defense using software defined networking. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN'12, pages 127–132, New York, NY, USA, 2012. ACM.

112 Douglas C. MacFarland and Craig A. Shue. The SDN shuffle: Creating a moving target defense using host-based software-defined networking. In *Proceedings of the Second ACM Workshop on Moving Target Defense*, pages 37–41. ACM, 2015.

113 Richard Skowyra, Kevin Bauer, Veer Dedhia, and Hamed Okhravi. Have no PHEAR: Networks without identifiers. In *Proceedings of the 2016 ACM Workshop on Moving Target Defense*, MTD'16, pages 3–14, New York, NY, USA, 2016. ACM.

114 S. Y. Chang, Y. Park, and A. Muralidharan. Fast address hopping at the switches: Securing access for packet forwarding in SDN. In *NOMS 2016–2016 IEEE/IFIP Network Operations and Management Symposium*, pages 454–460, April 2016.

115 H. Zhou, C. Wu, M. Jiang, B. Zhou, W. Gao, T. Pan, and M. Huang. Evolving defense mechanism for future network security. *IEEE Communications Magazine*, 53(4):45–51, April 2015.

116 Duohe Ma, Zhen Xu, and Dongdai Lin. *Defending Blind DDoS Attack on SDN Based on Moving Target Defense*, pages 463–480. Springer International Publishing, Cham, 2015.

117 Ankur Chowdhary, Adel Alshamrani, Dijiang Huang, and Hongbin Liang. Mtd analysis and evaluation framework in software defined network (mason). In *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*, SDN-NFV Sec'18, pages 43–48, New York, NY, USA, 2018. ACM.

118 J. B. Hong, S. Yoon, H. Lim, and D. S. Kim. Optimal network reconfiguration for software defined networks using shuffle-based online mtd. In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pages 234–243, Sept 2017.

119 Jafar Haadi Jafarian, Ehab Al-Shaer, and Qi Duan. *Formal Approach for Route Agility against Persistent Attackers*, pages 237–254. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

120 A. Aydeger, N. Saputro, K. Akkaya, and M. Rahman. Mitigating crossfire attacks using sdn-based moving target defense. In *2016 IEEE 41st Conference on Local Computer Networks (LCN)*, pages 627–630, Nov 2016.

121 Huiting Tan, Chaojing Tang, Chen Zhang, and Shaolei Wang. Area-dividing route mutation in moving target defense based on sdn. In Zheng Yan, Refik Molva, Wojciech Mazurczyk, and Raimo Kantola, editors, *Network and System Security*, pages 565–574. Springer International Publishing, Cham, 2017.

122 L. Zhang, Q. Wei, K. Gu, and H. Yuwen. Path hopping based SDN network defense technology. In *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*, pages 2058–2063, Aug 2016.

123 Zheng Zhao, Daofu Gong, Bin Lu, Fenlin Liu, and Chuanhao Zhang. SDN-based Double Hopping Communication against sniffer attack. *Mathematical Problems in Engineering*, vol. 2016 (13), 2016.

124 Duohe Ma, Cheng Lei, Liming Wang, Hongqi Zhang, Zhen Xu, and Meng Li. *A Self-adaptive Hopping Approach of Moving Target Defense to thwart Scanning Attacks*, pages 39–53. Springer International Publishing, Cham, 2016.

125 Ankur Chowdhary, Sandeep Pisharody, and Dijiang Huang. SDN based scalable MTD solution in cloud network. In *Proceedings of the 2016 ACM Workshop on Moving Target Defense*, MTD'16, pages 27–36, New York, NY, USA, 2016. ACM.

126 S. Debroy, P. Calyam, M. Nguyen, A. Stage, and V. Georgiev. Frequency-minimal moving target defense using software-defined networking. In *2016 International Conference on Computing, Networking and Communications (ICNC)*, pages 1–6, Feb 2016.

127 R. Callon E. Rosen, A. Viswanathan. Multiprotocol Label Switching Architecture. RFC 3031, RFC Editor, Jan 2001.

128 Kaleel Mahmood and Devu Manikantan Shila. Moving target defense for internet of things using context aware code partitioning and code diversification. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pages 329–330. IEEE, 2016.

129 Valentina Casola, Alessandra De Benedictis, and Massimiliano Albanese. *A Multi-Layer Moving Target Defense Approach for Protecting Resource-Constrained Distributed Devices*, pages 299–324. Springer International Publishing, Cham, 2014.

130 Ermanno Battista, Valentina Casola, Antonino Mazzeo, and Nicola Mazzocca. Siren: A feasible moving target defence framework for securing resource constrained embedded nodes. *International Journal of Critical Computer-Based Systems*, 4(4):374–392, 2013.

131 Matthew Sherburne, Randy Marchany, and Joseph Tront. Implementing moving target ipv6 defense to secure 6lowpan in the internet of things and smart grid. In *Proceedings of the 9th Annual Cyber and Information Security Research Conference*, CISR'14, pages 37–40, New York, NY, USA, 2014. ACM.

132 T. Preiss, M. Sherburne, R. Marchany, and J. Tront. Implementing dynamic address changes in Contiki OS. In *International Conference on Information Society (i-Society 2014)*, pages 222–227, Nov 2014.

133 K. Zeitz, M. Cantrell, R. Marchany, and J. Tront. Designing a micro-moving target ipv6 defense for the internet of things. In *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pages 179–184, April 2017.

134 K. Zeitz, M. Cantrell, R. Marchany, and J. Tront. Changing the game: A micro moving target ipv6 defense for the internet of things. *IEEE Wireless Communications Letters*, vol. 7(4), pp. 578–581, Aug. 2018.

135 Aljosha Judmayer, Georg Merzdovnik, Johanna Ullrich, Artemios G. Voyiatzis, and Edgar Weippl. A performance assessment of network address shuffling in IoT systems. In Roberto Moreno-D´ıaz, Franz Pichler, and Alexis Quesada-Arencibia, editors, *Computer Aided Systems Theory – EUROCAST 2017*, pages 197–204, Cham, 2018. Springer International Publishing.

136 Tommy Chin and Kaiqi Xiong. Mpbsd: A moving target defense approach for base station security in wireless sensor networks. In Qing Yang, Wei Yu, and Yacine Challal, editors, *Wireless Algorithms, Systems, and Applications*, pages 487–498, Cham, 2016. Springer International Publishing.

137 K. Andrea, A. Gumusalan, R. Simon, and H. Harney. The design and implementation of a multicast address moving target defensive system for internet-of-things applications. In *MILCOM 2017–2017 IEEE Military Communications Conference (MILCOM)*, pages 531–538, Oct 2017.

138 Massimiliano Albanese, Alessandra De Benedictis, Sushil Jajodia, and Kun Sun. A moving target defense mechanism for manets based on identity virtualization. In *2013 IEEE Conference on Communications and Network Security (CNS),* pages 278–286. IEEE, 2013.

139 Matt Richardson and Shawn Wallace. *Getting Started with Raspberry PI*. O'Reilly Media, Inc., 2012.

140 Raspberry Pi. Model b. 2015. URL: http://www.alliedelec.com/raspberry-piraspberry-pi-2-model-b/70465426, 2.

141 Raspberry Pi. Model b. *Raspberrypi. org. Saatavissa*. URL: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. Hakup¨aiva¨, 6:2018, 3.

142 8 devices. Carambola 2, 2018. URL: https://www.8devices.com/products/carambola-2

143 S. Groat, M. Dunlop, W. Urbanksi, R. Marchany, and J. Tront. Using an IPv6 moving target defense to protect the smart grid. In *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*, pages 1–7, Jan 2012.

144 J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. In *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005*, pages 364–369, April 2005.

145 A. Dunkels, B. Gronvall, and T. Voigt. Contiki – A lightweight and flexible operating system for tiny networked sensors. In *29th Annual IEEE International Conference on Local Computer Networks*, pages 455–462, Nov 2004.

146 Esraa M. Ghourab, Effat Samir, Mohamed Azab, and Mohamed Eltoweissy. Diversity-based moving-target defense for secure wireless vehicular communications. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 287–292. IEEE, 2018.

147 Vishwapathi Rao Tadinada. Software defined networking: Redefining the future of internet in IoT and cloud era. In *2014 International Conference on Future Internet of Things and Cloud (FiCloud),* pages 296–301. IEEE, 2014.

148 Fabrizio Granelli, Anteneh A. Gebremariam, Muhammad Usman, Filippo Cugini, Veroniki Stamati, Marios Alitska, and Periklis Chatzimisios. Software defined and virtualized wireless access in future wireless networks: Scenarios and standards. *IEEE Communications Magazine*, 53(6):26–34, 2015.

149 Zhijing Qin, Grit Denker, Carlo Giannelli, Paolo Bellavista, and Nalini Venkatasubramanian. A software defined networking architecture for the internet-of things. In *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pages 1–9. IEEE, 2014.

150 Alejandro De Gante, Mohamed Aslan, and Ashraf Matrawy. Smart wireless sensor network management based on software-defined networking. In *2014 27th Biennial Symposium on Communications (QBSC),* pages 71–75. IEEE, 2014.

151 Tie Luo, Hwee-Pink Tan, and Tony QS Quek. Sensor openflow: Enabling software defined wireless sensor networks. *IEEE Communications Letters*, 16(11):1896–1899, 2012.

152 Amr El-Mougy, Mohamed Ibnkahla, and Lobna Hegazy. Software-defined wireless network architectures for the internet-of-things. In *2015 IEEE 40th Local Computer Networks Conference Workshops (LCN Workshops),* pages 804–811. IEEE, 2015.

153 Slavica Tomovic, Kenji Yoshigoe, Ivo Maljevic, and Igor Radusinovic. Software defined fog network architecture for IoT. *Wireless Personal Communications*, 92(1):181–196, 2017.

154 Zhigang Wen, Xiaoqing Liu, Yicheng Xu, and Junwei Zou. A restful framework for internet of things based on software defined network in modern manufacturing. *The International Journal of Advanced Manufacturing Technology*, 84 (1–4):361–369, 2016.

155 Manisha Chahal, Sandeep Harit, Krishn K. Mishra, Arun Kumar Sangaiah, and Zhigao Zheng. A survey on software-defined networking in vehicular ad hoc networks: Challenges, applications and use cases. *Sustainable Cities and Society*, Vol. 35, Pages 830–840. 2017.

156 Ian Ku, You Lu, Mario Gerla, Rafael L. Gomes, Francesco Ongaro, and Eduardo Cerqueira. Towards software-defined vanet: Architecture and services. In *2014 13th annual Mediterranean ad hoc networking workshop (MED-HOC-NET)*, pages 103–110. IEEE, 2014.

157 Charles Perkins, Elizabeth Belding-Royer, and Samir Das. Ad hoc on-demand distance vector (aodv) routing. Technical report, 2003.

158 Thomas Clausen and Philippe Jacquet. Optimized link state routing protocol(olsr). Technical report, 2003.

159 Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers. *In ACM SIGCOMM Computer Communication Review*, volume 24, pages 234–244. ACM, 1994.

160 Baihong Dong, Weigang Wu, Zhiwei Yang, and Junjie Li. Software defined networking based on-demand routing protocol in vehicle ad hoc networks. In *2016 12th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN),* pages 207–213. IEEE, 2016.

161 David B. Johnson, David A. Maltz, Josh Broch, et al. Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks. *Ad Hoc Networking*, 5:139–172, 2001.

162 A. Kott, A. Swami, and B. J. West. The internet of battle things. *Computer*, 49 (12):70–75, Dec. 2016.

163 Mohammad Ashiqur Rahman, Ehab Al-Shaer, and Rakesh B. Bobba. Moving target defense for hardening the security of the power system state estimation. In *ACM Workshop on Moving Target Defense (MTD)*, pages 59–68, 2014.