CHAPTER 6

Security of Distributed Intelligence in Edge Computing: Threats and Countermeasures

Mohammad S. Ansari, Saeed H. Alsamhi, Yuansong Qiao, Yuhang Ye, and Brian Lee

Abstract Rapid growth in the amount of data produced by IoT sensors and devices has led to the advent of edge computing wherein the data is processed at a point at or near to its origin. This facilitates lower latency, as well as data security and privacy by keeping the data localized to the edge node. However, due to the issues of resource-constrained hardware and software heterogeneities, most edge computing systems are prone to a large variety of attacks. Furthermore, the recent trend of incorporating intelligence in edge computing systems has led to its own security issues such as data and model poisoning, and evasion attacks. This chapter presents a discussion on the most pertinent threats to edge intelligence.

M. S. Ansari (✉) • Y. Qiao • Y. Ye • B. Lee
Software Research Institute, Athlone IT, Athlone, Ireland
e-mail: mansari@ait.ie; ysqiao@research.ait.ie; yye@research.ait.ie; blee@ait.ie

S. H. Alsamhi
IBB University, Ibb, Yemen
e-mail: saeedalsamhi@gmail.com

T. Lynn et al. (eds.), *The Cloud-to-Thing Continuum*, Palgrave Studies in Digital Business & Enabling Technologies,
https://doi.org/10.1007/978-3-030-41110-7_6

Countermeasures to deal with the threats are then discussed. Lastly, avenues for future research are highlighted.

Keywords Edge AI • Edge computing • Distributed intelligence • Federated learning • Threats to Edge AI

6.1 Edge Computing: Threats and Challenges

As discussed in Chap. 1, edge computing refers to data processing at or near the point of its origin rather than onward transmission to the fog or cloud. The 'edge' is defined as the network layer encompassing the smart end devices and their users, and is identified by the exclusion of cloud and fog (Iorga et al. 2018). For instance, a smartphone is the edge between body things and the cloud, and a gateway in a smart home is the edge between home things and the cloud (Shi et al. 2016).

Although edge computing brings a lot of advantages, and is being used in a variety of scenarios, it is not without its share of security threats and challenges. In fact, the following factors work towards expanding the attack surface in the case of edge computing:

Hardware Constraints: Since most edge computing hardware (edge devices, and even edge servers) have lower computational power and storage capacity as compared to a fog or cloud server, they are incapable of running dedicated attack prevention systems like firewalls, and are therefore more vulnerable to attacks.

Software Heterogeneities: Most devices and servers operating in the edge layer communicate using a large variety of protocols and operating systems without a standardized regulation. This makes the task of designing a unified protection mechanism difficult.

Most of these threats are exacerbated due to design flaws, implementation bugs, and device misconfigurations in the edge devices and servers (Xiao et al. 2019). Also, the lack of full-fledged user interfaces in many edge devices often makes it impossible to discern an ongoing/transpired attack.

In light of the above, understanding the security threats (and defenses) in edge computing assumes utmost importance. This section presents an overview of the state-of-the-art in the security threats and countermeasures employed in edge computing.

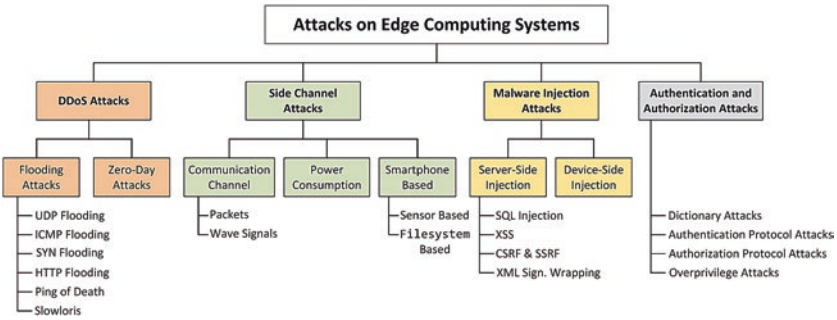

Fig. 6.1 Different types of attacks against edge computing systems (Xiao et al. 2019)

As depicted in Fig. 6.1, most attacks on edge computing infrastructure may be placed in one of the following four categories: DDoS attacks, side-channel attacks, malware injection attacks, and authentication and authorization attacks (Xiao et al. 2019). Each of these attacks and the countermeasure devised to deal with the corresponding attacks are discussed next.

6.1.1 DDoS Attack

In this type of attack, the goal of the adversary is to engage all the resources and bandwidth available at the target in order to prevent legitimate users from using the victimized system. In a typical DDoS attack, the attacker persistently sends a huge number of packets to the target (also referred to as 'flooding') thereby ensuring that all the resources of the target are exhausted in handling the malicious packets, and therefore genuine requests cannot be processed. Such attacks assume greater importance in the edge computing paradigms as they are computationally less powerful (than cloud servers), and therefore cannot run strong defense systems. Such attacks may be further categorized as UDP flooding attacks, ICMP flooding, SYN flooding, ping of death (PoD), HTTP flooding, and Slowloris (Xiao et al. 2019). Apart from the flooding attacks, another type of DDoS attack is a zero-day attack in which an attacker finds and utilizes a still-unidentified vulnerability in the target system to cause system shutdown.

Defenses and Countermeasures: Most potent solutions against flooding attacks utilize the detect-and-filter technique. The detection of malicious

flooding packets may either be on a per-packet basis wherein each individual packet is inspected and discarded if deemed to be suspicious, or on a statistical basis wherein malicious packets are identified using parameters like packet entropy or by employing machine learning tools. Countering zero-day attacks on edge computing hardware is more difficult due to the unavailability of original source codes for the programs running on the machine, and also due to the fact that in many cases the software comes embedded in a firmware and is not amenable for inspection.

6.1.2 Side-Channel Attacks

These attacks operate by first capturing publicly available, non-privacy-sensitive information pertaining to the target (also called the side-channel information), and then inferring the private and protected data from this information by exploiting the correlations that are inherently present between the public and the private information. Typical examples of such attacks include capturing communication signals (e.g. packets or wave signals) to leak user's private data, monitoring the power consumption of edge devices to reveal usage patterns, and targeting the filesystem (e.g. the */proc* filesystem in Android) and sensors (e.g. microphone, camera) on end devices like smartphones.

Defenses and Countermeasures: Due to their passive nature, side-channel attacks are difficult to defend against. Some commonly suggested defense mechanisms include data perturbation and differential privacy. The most popular data perturbation algorithm is *k*-anonymity which modifies the identifier information in the data prior to publishing its sensitive attributes. Lastly, it is important to note that ironically most defense mechanisms are themselves vulnerable to side-channel attacks (Xiao et al. 2019).

6.1.3 Malware Injection Attacks

The infeasibility of installing a full-fledged firewall on resource-constrained edge devices makes them vulnerable to malware injection attacks, wherein an attacker stealthily installs malicious programs in a target system. Such malware injection may either be performed at the edge server or the edge device(s). Server-side injection attacks can further be divided into four types: SQL injection, cross-site scripting (XSS), XML signature wrapping, and Cross-Site Request Forgery (CSRF) & Server-Site Request Forgery

(SSRF) (Xiao et al. 2019). Device-side injection attacks typically target the firmware of the end devices.

In a SQL injection attack, the attacker aims to destroy the backend database by sending carefully crafted SQL queries containing malicious executable codes. In a XSS attack, the adversary injects malignant HTML/JavaScript codes into the data content which may be accessed and executed by a server leading to its compromise. A CSRF attack is one in which the edge server is tricked into executing malicious programs embedded in web applications, and a SSRF attack is carried out by compromising and using an edge server to alter the internal data and/or services. Lastly, an XML signature wrapping attack works by intercepting and modifying a XML message, and re-transmitting it to a target machine in order to run tainted code.

Defenses and Countermeasures: To counter the server-side injection attacks, the detect-and-filter technique has been shown to be the most promising. Defense mechanisms against injection attacks generally rely on static analysis for malicious code detection and fine-grained access control. Research on devising means to mitigate firmware modification is also being carried out for prevention of such attacks.

6.1.4 Authentication and Authorization Attacks

The authentication and authorization processes in edge computing systems may also be susceptible to attacks. Such attacks may be put into four different categories: dictionary attacks, attacks targeting vulnerabilities in authentication mechanisms, attacks exploiting susceptibilities in authorization protocols, and over-privileged attacks (Xiao et al. 2019). Dictionary attacks employ a credential/password dictionary to get past the authentication systems. Attacks targeting vulnerabilities in authentication mostly work by utilizing loopholes in the WPA/WPA2 security protocols. Authorization based attacks exploit the logical weaknesses or design flaws that may exist in authorization protocols used by the edge computing systems. In over-privileged attacks, the attacker tricks the victim system into assigning higher (than required) access rights to an app or device, which can then be used to perform malicious activities inside the network.

Defenses and Countermeasures: The most potent defense against dictionary attacks is the addition of one more layer of authentication (typically known as two-factor authentication). To counter the attacks which target authentication protocols, two common approaches are enhancing

the security of the communication protocols, and hardening the cryptographic implementation. The OAuth 2.0 protocol is the best defense against authorization attacks, and has been proven to be theoretically secure. To counter the over-privileged attacks, the most effective solution involves strengthening the permission models for the operating systems running on edge devices.

Most of the security threats and challenges, along with the associated countermeasures, discussed above pertain to edge computing systems which are configured as passive data aggregation and processing nodes with little to no intelligence built into them. However, the recent trend of incorporation of intelligence (in the form of inference generation, and even on-device training, in the context of machine learning) into the edge nodes/devices, brings its own share of issues and challenges, and the need for specialized defenses and countermeasures.

This chapter aims to highlight the threat landscape for the scenario where edge devices are becoming smarter with the inclusion of machine learning. Therefore, the remainder of the chapter focuses on the techniques for incorporation of intelligence into edge computing systems, the security threats associated with such systems, and the pertinent countermeasures and defenses that have been devised against attacks on edge intelligence. Section 6.2 presents a discussion on the need for, and the techniques to bring intelligence to the edge computing systems. Security threats targeted towards intelligent edge systems are highlighted in Sect. 6.3 (For a quick summary, please refer to Table 6.1). Techniques that have been developed to defend against the threats, and mitigate the attacks on edge computing systems are discussed in Sect. 6.4. Section 6.5 contains a discussion on future research directions in the field of intelligent edge computing. Section 6.6 presents concluding remarks.

6.2 Edge Intelligence

The incorporation of artificial intelligence into the constituents of edge layer is referred to as *Edge AI*. The two biggest advantages of Edge AI are briefly discussed below.

Faster Inference: For applications which utilize a pre-trained machine learning model to output classifications or predictions, processing data at the edge leads to faster results. This is primarily due to the elimination of the data transfer time between the edge and the cloud.

Table 6.1 Security threats to edge computing systems, defense mechanisms, and assets targeted by the different attacks

Attack	*Type*	*Sub-type*	*Ref.*	*Defense*	*Asset targeted*
DDoS attack	Flooding based	UDP flooding	Xiaoming et al. (2010)	Detect-and-filter	• Network infrastructure
		ICMP flooding	Udhayan and Anitha (2009)	• Per packet based detection	• Virtualization infrastructure
		SYN flooding	Bogdanoski et al. (2013)	• Statistics-based detection	
		HTTP flooding	Dhanapal and Nithyanandam (2017)		
		Ping of death	Sonar and Upadhyay (2014)		
		Slowloris	Damon et al. (2012)		
	Zero-day		NIST (2010)	Code-level vulnerability identification	
Side-channel attack	Exploit communication channels	Packets	Chen and Qian (2018)	Data perturbation	• User data
		Wave signals	Enev et al. (2011)	• *k*-anonymity	• User privacy
				• Differential privacy	
	Exploiting power consumption data		Örs et al. (2003)	Restricting access to side-channels	
	Target smart devices	OS based	Zhou et al. (2013)		
		Sensor based	Chen et al. (2018b)		
Malware injection attacks	Server-side injection	SQL injection	Anley (2002)	Detect-and-filter	• Edge server
		XSS	Cisco (2016)		• Edge devices
		CSRF & SSRF	Costin (2018)		
		XML signature wrapping	McIntosh and Austel (2005)		
	Device-side injection	Buffer overflow	Greenberg (2017)	Code-level analysis	

(*continued*)

Table 6.1 (continued)

Attack	*Type*	*Sub-type*	*Ref.*	*Defense*	*Asset targeted*
Authentication and authorization attacks	Dictionary attack		Nakhila (2015)	Two-factor authentication	• Edge server • Virtualization infrastructure • Edge devices
	Authentication protocol attack		Vanhoef and Piessens (2018)	Hardening authentication protocols • Enhance security of protocol • Secure cryptographic implementation	
	Authorization protocol attack		Chen (2014)	Hardening authorization protocols • OAuth 2.0	
	Over-privileged attacks		Sun and Beznosov (2012)	Strengthening permission models for mobile OS	

Data Locality: Since most of the data processing and inference is performed it the edge layer, the data actually never leaves this layer (and is not sent to the fog/cloud). Such data locality is of paramount importance in safeguarding user privacy in applications like health monitoring, indoor localization, etc. Further, keeping the data on or near the source, and not transferring it to the cloud (which may be in a different country), alleviates regulatory/legal issues pertaining to the data.

Although the advantage of faster inference with the data remaining localized is interesting, the resource constraints in most constituents of the edge layer dictate that specialized techniques have to be employed for performing inference and training in Edge AI.

6.2.1 *Lightweight Models for Edge AI*

The first case is where an edge computing node is only used for inference using a pre-trained model. In such cases, the emphasis is to build lightweight models capable of running in resource constrained environments. This discussion will focus on image processing models because a major portion of available research on light models for Edge AI deals with computer vision. This is driven by the success of Convolutional Neural Networks (CNN) for image recognition and classification tasks, albeit with huge computational requirements. AlexNet was the first CNN variant which employed a technique called Group Convolution to reduce the number of parameters, and resulted in a 240 MB sized model (Krizhevsky et al. 2012). Xception used a more stringent version of group convolution to further reduce the number of model parameters (88 MB model size) (Chollet 2017). GoogleNet managed to reduce the parameter size to 27 MB while maintaining the accuracy (Szegedy et al. 2015). However, the breakthrough which enabled CNN variants to be used on edge devices was MobileNet (Howard et al. 2017), which required approximately 8–9 times less computation than standard CNN, and had model size of 16 MB (Howard et al. 2017). MobileNet*V2* further provided a performance improvement while reducing the model size to 14 MB (Sandler et al. 2018). SqueezeNet is even more efficient, and is capable of providing AlexNet level accuracy with only 5 MB of parameters (Iandola et al. 2016), which is a sufficiently small sized model for deployment on low-complexity embedded hardware like Raspberry Pi.

6.2.2 *Data and Model Parallelism*

For cases where the edge computing nodes are to be used for training as well, techniques like data parallelism and model parallelism are employed.

Data Parallelism: In data parallelism, the training dataset is divided into non-overlapping partitions and fed to the participating nodes. Figure 6.2(a) depicts the data parallelism applied to a group of three machines. All nodes train the complete model using a subset of data. The advantage is that the training task is performed at multiple nodes concurrently (for different data sub-sets). Specialized algorithms like Synchronous Stochastic Gradient Descent (Sync-SGD) (Das et al. 2016), and Asynchronous Stochastic Gradient Descent (Async-SGD) (Zhang et al. 2013) have been devised to ensure timely and efficient update of the global weights and parameters of the model.

Model Parallelism: In model parallelism, the ML model is divided into partitions and each participant node is responsible for maintaining one partition. Figure 6.2(b) depicts the model parallelism applied to a group of four machines. Designing the model partitions is non-trivial and NP-complete in this case, as the participating machines may have different storage, computing, and networking capabilities (Dean et al. 2012). Further, dividing the training dataset is also not straightforward in this case, as the logical partitions have to be decided in accordance with the partition scheme of the input layer.

To reduce the communication of a large number of parameters between participating devices, *model compression* is used. It has been demonstrated that quantizing the parameter bitwidth from 32 bits to 8 bits does not

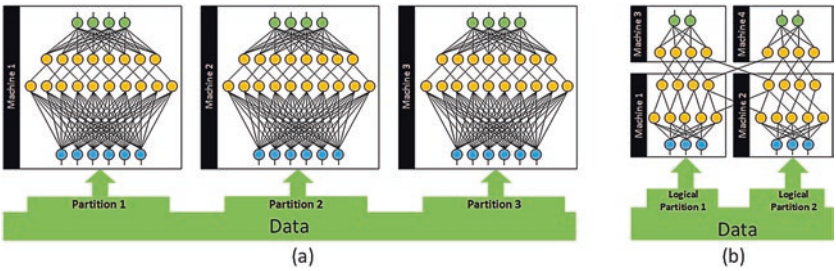

Fig. 6.2 (**a**) Data parallelism and (**b**) model parallelism

impact the accuracy of CNN-like architectures significantly (Cheng et al. 2017). Further, reducing the communication overhead by quantizing the gradients (computed using Stochastic Gradient Descent) is referred to as *Gradient Compression* or *Gradient Quantization*.

6.2.3 *Federated Learning*

Data collected by a lot of devices may not be amenable for sharing over a cloud due to reasons of privacy. Examples include data collected by health monitoring devices, CCTV recordings, etc. For such cases, a distributed ML technique called Federated Learning (FL) has been proposed (Konečný et al. 2016), which enables smart devices to collaboratively learn a shared prediction model while keeping all the training data on device. This effectively decouples the learning process from the need to store the data centrally, and goes beyond the use of pre-trained models to make predictions on mobile devices by bringing model training to the device. As shown in Fig. 6.3, FL works by first downloading the current model to an edge device. Thereafter, the model is updated using locally stored data, and updates are forwarded to a central server where they undergo a

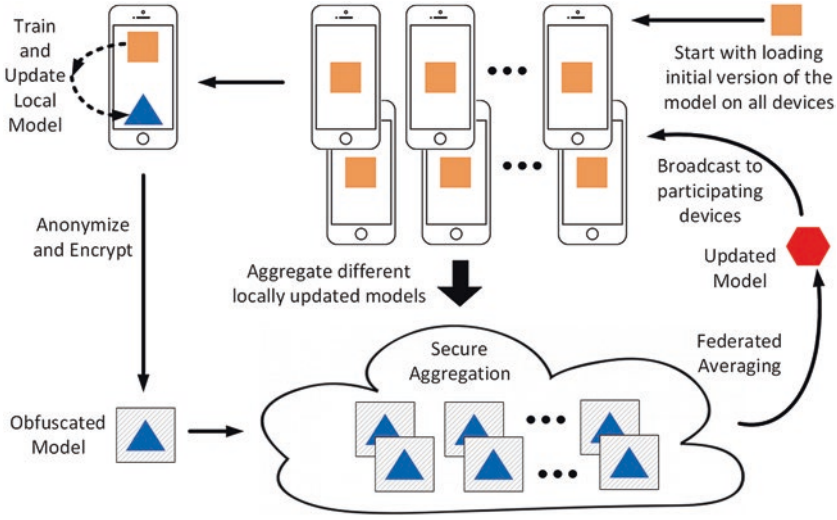

Fig. 6.3 Federated learning over multiple smartphones

Federated Averaging with the updates from other users. Since the user data never leaves the device, and individual updates are not stored in the cloud, data security and privacy is ensured.

The updates in this case are not simple gradient updates as in the case of conventional distributed ML models. Instead, high-quality updates containing much more information than just the changes in gradients are computed, compressed, and sent for processing. This ensures convergence with reduced communication (up to 100 times (Konečný et al. 2016)) between the edge device and the central server. Scheduling algorithms are used to ensure that training happens only when the device is idle, charging, and on a free wireless connection, so there is no degradation in the end-user experience. With most flagship phones nowadays coming with a dedicated AI chip, there are estimated to be approximately two billion smartphones with underutilized processing capability. Federated Learning can leverage this enormous pool of computing resources to improve existing models, or to train new ones from scratch.

The distribution of intelligence over a multitude of end devices is therefore slated to bring significant improvements in the way conventional IoT devices function. However, this distribution of intelligence to the edge nodes also opens up a plethora of security issues which are discussed next.

6.3 Threats to Edge AI

Despite their widespread usage by virtue of the advantages they offer, Edge AI paradigms are not without their share of limitations and points of concerns. Incorporating intelligence in the edge layer is a double edged sword in the sense that although the impact of a potential attack is limited to a localized environment, the less potent security protocols on the resource-constrained edge hardware make them more vulnerable to attacks. The situation is further aggravated by the casual attitude of human operators responsible for the configuration and maintenance of the edge devices. For instance, a survey of 439 million households using WiFi networks showed that approximately 50% of them were unsecured, and of the remaining, 80% have their router still configured with the default passwords (Shi et al. 2016). The figure is even poorer for public WiFi hotspots, with 89% of them being unsecured or poorly configured (Shi et al. 2016). Furthermore, updating or re-configuration of the security software on edge devices is non-trivial because there may be legacy devices for which support has ended, or the constrained hardware resources available on the

device may present restrictions on the authentication protocols that could be run on the device. Moreover, the heterogeneous nature of the edge networks means that there can be no uniform security policy. Lastly, microservers used in the edge computing environment lack the hardware protection mechanisms available on commodity servers (Roman et al. 2018).

A discussion on the threats to Edge AI systems can be divided into two distinct cases: threats to Edge AI used for inference, and threats to Edge AI used for learning/training (as in Federated Learning). Each of these scenarios are discussed separately below. It needs to be mentioned that in the discussion that follows, it is considered that the intelligence is located in the edge device. However, this is not a restrictive scenario. In fact, the attacks and countermeasure discussed below are equally relevant in the case where the machine intelligence is located in an edge server or a gateway.

6.3.1 Threats to Edge AI for Inference

The vast majority of Edge AI deployments at present are used for inferencing based on pre-trained models. This is suitable for edge devices due to the limited computing resources they offer. As discussed before, there has been progress in model compression that allow high performance models (e.g. SqueezeNet) to be run in resource-constrained environments. In such a standalone environment, where the edge devices use the pre-trained model independently, the most probable attack is the feeding of adversarial examples to the model thereby causing the model to output incorrect predictions. Such attacks are referred to as Evasion Attacks and are discussed next.

6.3.2 Evasion Attacks

The susceptibility of machine learning models to adversarial samples, which essentially are carefully perturbed inputs that look and feel exactly the same as their untampered counterparts to a human, is well documented (Biggio and Roli 2018). Although it may seem that adversarial examples are available only for image recognition models (Kurakin et al. 2016), the earliest reported instance of such an attack is for a machine learning based email spam filter, wherein it was shown that linear classifiers could be tricked easily by carefully crafted changes in the text (Dalvi et al.

2004). It is still not proven why adversarial samples work, but a commonly accepted hypothesis, called the tilted boundary hypothesis, asserts that since the model can never fit the data perfectly (at least theoretically), there would always be adversarial pockets of inputs existing between the classifier boundary and the actual sub-manifold of sampled data (Szegedy et al. 2013). Since the models devised to be used in the low resource environments of edge computing are compressed variants of bigger, deeper, and more robust models, these are generally more prone to such adversarial attacks. Evasion attacks can be of different types: Gradient based, Confidence-score based, Hard Label based, Surrogate model based, and Brute-force attacks (Moisejevs 2019).

Gradient based attacks require access to the model gradients (and thus belong to the category of Whitebox attacks). Theoretically, such attacks are the most potent as the attacker may use the model gradients to gain insights into the working of the model, and can then mathematically optimize the attack. This approach is the most probable one to target hardened models, as it has been shown that if an adversary has access to the model gradients, it is *always* possible to generate adversarial samples irrespective of the robustness of the model (Carlini and Wagner 2017). Some examples of such attacks include Elastic-Net attack based on L_1 norm (Chen et al. 2018a), an L_2 Norm based attack (Carlini and Wagner 2017), and an L_∞ Norm based attack (Madry et al. 2017).

Confidence-score based adversarial attacks utilize the output confidence score to get estimates of the gradients of the model. The adversary may then use these estimated gradients to orchestrate an attack similar to the gradient based attack. Since this approach does not require any information about the composition of the model, this attack may be classified as a Blackbox attack. Examples include the Zeroth Order Optimization based attack (Chen et al. 2017a), the Natural Evolutionary Strategies (NES) based attack (Ilyas et al. 2018), and the Simultaneous Perturbation Stochastic Approximation (SPSA) based attack (Uesato et al. 2018).

Label based attacks rely on estimating the gradients by using the hard labels generated by the model. Since only the label information is required by the adversary, such attacks are generally simple to implement, and require little hyperparameter tuning. Boundary Attack is the most powerful attack in this category. It works by starting from a large adversarial perturbation and seeks to incrementally reduce the perturbation while staying adversarial (Brendel et al. 2017).

Surrogate model based attacks first try to build a replica of the target model. If the internals of the target model are not known, the adversary can reverse engineer the structure of the model by repeatedly querying the target model and observing the input-output pairs. If the target model is not available for querying, then the attacker can start by guessing the architecture in the case of the model being applied for a standard machine learning problem like image classification (Moisejevs 2019). Thereafter, the gradient based attack can be fine-tuned on this surrogate model, and then used on the actual model.

Lastly, Brute-force attacks, as the name implies, work by generating adversarial examples by resorting to transformations, perturbations and addition of noise to the data samples. Such attacks do not rely on mathematical optimization, and therefore require no knowledge of the model. Such an approach is generally used by adversaries who have access to large computational resources, and do not have a timeline for the success of their attacks.

6.3.3 Privacy Attacks

The previous section discussed the issues pertaining to evasion attacks wherein the goal of the attacker is to cause the model to output incorrect predictions. However, there is another class of attacks, known as Privacy Attacks, which aim to siphon off valuable information from the data used by the model. For instance, an adversary may be interested in knowing whether a certain person is enrolled in a healthcare program. There are several other examples of such private information which an attacker may want to unravel: credit card details, location information, and household energy consumption. While the risk with disclosure of credit card information is obvious, the availability of location and energy usage information of a person can inform the attacker about when the person is away for a vacation (consequently leaving his house unattended). There are two broad categories of such privacy attacks on machine learning systems:

Membership Inference Attacks: This is the case when the adversary has one or more data points, and wants to ascertain whether the data points were part of the training set or not (Shokri et al. 2017). For instance, an attacker might want to find out whether a given person X is included in a critical illness list in the healthcare records of a state. Such attacks are increasingly being targeted towards recommender systems, wherein the

training dataset may contain information such as gender, age, ethnicity, location, sexual orientation, immigration status, political affiliation, net worth and buying preferences. An attacker who knows a few pieces of information from these may be able to expose other details using membership inference. A detailed study of such attacks has been carried out (Truex et al. 2019), which concluded that several factors affected the potency of membership inference attacks. Firstly, the model becomes more vulnerable with increase in the number of classes. Also, the choice of the algorithm for training is also an important factor. Algorithms whose decision boundaries are not significantly impacted by an individual training sample are less vulnerable.

Model Inversion Attacks: Such attacks, also known as Data Extraction attacks, work by extracting an average representation of each of the classes the target model was trained on. For instance, a model trained for facial recognition may be attacked in the following manner. First, a base image is chosen based upon the physical characteristics (age, gender, ethnicity) of the person whose image is to be extracted from the model. Then the attacker can repeatedly query the target model with different modifications in the base image, until a desired confidence level is reached. It has been shown that the final image in such an attack scenario can be fairly demonstrative of the face of the person concerned (Fredrikson et al. 2015). With the increasing integration of ML based face recognition systems in modern day security and surveillance setups including the ones at airports, such attacks may lead to the divulgence of private and sensitive information like photographs, visa and passport details, travel itineraries, and much more. In another instance, it has been demonstrated that it is possible to extract credit card details and social security numbers from a text generator trained on private data (Carlini et al. 2019).

6.3.4 Threats to Edge AI for Training

This section deals with the threats that are pertinent for Edge AI systems which are used for performing both machine learning training and inference. Firstly, the convergence guarantee of the federated learning algorithms has not still been theoretically established (Ma et al. 2019). Only approximate convergence may be guaranteed, and that too requires some unrealistic assumptions: (1) training data is shared across devices or distributed amongst the participating devices in an independent and

identically distributed (IID) manner, and (2) all participating devices are involved in communication of updates for each round.

Secondly, in the federated learning scenario, an adversary can take control over one or more participating devices to inject spurious and arbitrary updates in order to manipulate the training process. This is generally referred to as *model poisoning* or *logic corruption*. Also, a malicious intruder may also compromise the training data in order to adversely affect the training process. This is commonly known as *data poisoning*, and may be in the form of either the manipulation of the labels in the training data, or the modification of the input itself. It has been shown that an adversarial participant can infer properties associated with a subset of training data (Bagdasaryan et al. 2018). Also, there may exist eavesdroppers on the broadcast link used by the centralized server to communicate the intermediate model state to the participants. Another way of classifying the poisoning attacks on Edge AI systems can be based on the characteristic that is targeted to be compromised. For instance, attacks targeting the *availability* of the system generally work by injecting a lot of spurious data into the training set, thereby ensuring that whatever classification boundary the model learns becomes useless. It has been shown that a 3% poisoning of the dataset can lead to more than 10% drop in accuracy (Steinhardt et al. 2017). Such attacks are the ML counterparts to the conventional Denial-of-Service attacks. Another class of attacks do not aim to affect the availability of the ML system, and instead target the *integrity* of the system. Such attacks are more sophisticated than availability attacks, and leave the classifier functioning exactly as it should, but with one or more backdoor inputs embedded into the model. These backdoor inputs cause the classifier to output incorrect predictions thereby compromising the integrity of the model. An example of such a backdoor input is a spam email checking scenario wherein an attacker teaches a model that if a certain string is present in the input, then that input is to be classified as benign (Chen et al. 2017b).

Further, although the concept of federated learning is appealing, it remains to be seen how it performs with scaling up. Several practical issues are expected to creep up when the FL systems are scaled up to involve a huge number of devices: limited device storage, unreliable connectivity, and interrupted execution. Moreover, it is still unknown whether a significant increase in the number of participating devices would translate to better accuracy and/or faster convergence of the model.

There can be another way of looking at the threats that may affect Edge AI. Typically, an Edge AI system is composed of three major components: network, services, and devices. The network (generally wireless network) may be susceptible to DoS and man-in-the-middle attacks, as well as prone to disruptions by a rogue node or gateway. The services running on the nodes may be infiltrated to cause privacy leakage, privilege escalation, and service manipulation. Lastly, the edge devices may themselves be prone to physical damage, as well as data poisoning.

6.4 Countering the Threats to Edge AI

This section presents a discussion on the techniques available for dealing with the threats against Edge AI. Since the threats could be against the data, the model, or even the entire system (e.g. Federated Learning), the following discussion is structured accordingly. At the onset, it needs to be mentioned that no available countermeasure can be guaranteed to completely eliminate the threats to Edge AI systems, and it is by a judicious mix of the defense techniques that we can hope for a reasonable safe system.

6.4.1 Defenses against Data Poisoning

In a data poisoning attack on a machine learning system, the adversary injects malicious samples into the training pool. These tainted data samples are typically significantly different from the benign data points, and are therefore 'outliers.' The process of outlier detection (also known as anomaly detection or data sanitization) aims to identify and eliminate such outliers *before* the training process (Paudice et al. 2018). The anomaly detection process is obviously ineffective if the poisoned samples were introduced into the training dataset before the filtering rules were created. Further, if the attacker is able to generate data poison samples which are very similar to the pristine samples ('inliers'), then this line of defense breaks down. Another variant of the anomaly detection approach is the use of micromodels (Cretu et al. 2008). The Micromodel approach was first proposed for use in network intrusion detection datasets, wherein multiple micromodels were generated by training the classifier on non-overlapping slices of the training sets (micromodels of the training set). A majority voting scheme was then used on the micromodels to ascertain which of the training slices were corrupted by poisoning. The institution

behind this approach is that network attacks are generally of a low time duration, and can only affect a few training slices.

Another commonly used defense technique is to analyze the effect of a new sample on the model's accuracy before actually including that sample in the training set. For a tainted data sample used as a test sample, the model's accuracy would degrade. Reject on Negative Impact (RONI) (Nelson et al. 2009), and target-aware RONI (tRONI) (Suciu et al. 2018) are defensive methods that use this approach. The RONI defense has been demonstrated to be extremely successful against dictionary attacks on email spam filters, identifying 100% of malicious emails without flagging any benign emails. However, RONI fails to mitigate targeted attacks because the poison instances in such cases might not individually cause a significant performance drop. Target-aware RONI was then proposed as a targeted variant which is capable of identifying instances that distort the target classification significantly.

A perturbation approach has also been employed for anomaly detection (Gao et al. 2019). STRong Intentional Perturbation (STRIP) intentionally perturbs the incoming data samples, for instance by superimposing different patterns on sample images, and observes the randomness of the predicted classes for the perturbed inputs. It is expected that a benign classifier would be affected significantly by the perturbations. A low entropy in the classes predicted by the model defies the input-dependence property of a pristine model and implies the presence of a tainted input.

Another method known as TRIM has been proposed for regression learning. It estimates the parameters iteratively, while employing a trimmed loss function to remove samples which lead to large residuals. It has been demonstrated that TRIM is able to isolate most of the poisoning points and learn a robust regression model (Jagielski et al. 2018).

Lastly, even after significant strides in automated anomaly detection, the role of human factors in identifying malicious data samples cannot be completely eliminated. Human-in-the-loop approach works by focusing the attention of human data analysts on outliers which cause an unwarranted boundary shift in a classifier model (Mei and Zhu 2015).

6.4.2 Countering Adversarial Attacks

Defenses against evasion attacks may be put into two broad categories: formal methods and empirical approaches. Formal methods are purely mathematical in nature, and work by testing the model on all possible

adversarial samples which can be generated within the allowable limits of perturbation. While this approach leads to virtually impenetrable models, the method is not amenable to most present day applications of machine learning due to its high requirement of computational resources. For instance, applying formal methods to a model working with image inputs would mean generating all adversarial images (within a certain noise range), feeding them to the model and verifying whether the output is as intended. Therefore, this class of countermeasures is still more theoretical than practical.

Empirical defenses, on the other hand, rely on experiments to ascertain the effectiveness of a defense mechanism. There are several defense strategies which can be employed. Adversarial training refers to retraining of the model with adversarial samples included in the training set after including their correct labels. It is expected that this will ensure that the model learn to ignore the noise and focus on the more evident features in the entire training set. A technique called Ensemble Adversarial Training (EAT) has been proposed that augments training data with perturbations transferred from other models, thereby making the model more robust (Tramèr et al. 2017). Cascade adversarial training, which transfers the knowledge of the end results of adversarial training on one model, to other models has been proposed to enhance the robustness of models (Na et al. 2017). A robust optimization based approach for identifying universally applicable, reliable training methods for neural networks has also been proposed (Madry et al. 2017).

Other commonly used technique to defend models against evasion attacks is input modification. In this case, an input sample, prior to being fed to the model, is passed through a sanitizing system to remove the adversarial noise, if any. Examples of such methods include denoising approaches like autoencoders and high level representational denoisers, JPEG compression, pixel deflection, and general basis function transformations (Moisejevs 2019). Lastly, there is an interesting NULL class approach (Hosseini et al. 2017), in which the classifier is trained to output a NULL class for inputs which it considers as adversarial.

6.4.3 Hardening Federated Learning Systems

Since the process of training, aggregation and model updating is spread over the client, server, and the network in a federated learning system, all the three segments need hardening against potential adversaries. Privacy

protection at the client side may be ensured by adding perturbations (noise) to the updates (Ma et al. 2019). The more sensitive attributes in the update can be obscured by using differential privacy techniques (Dwork et al. 2006).

The server side can be made more robust by incorporating Secure Multi-Party Computation (SMC) which ensures that individual updates are rendered uninspectable at the server (Rosulek 2017). A secure aggregation protocol can be employed that uses cryptographic techniques so a coordinating server can only decrypt the average update if a certain number of users have participated, and no individual update can be inspected before averaging. A variety of other specialized approaches have also been employed to safeguard user privacy. These include, but are not limited to, de-identification schemes like anonymization, and cryptographic techniques like homomorphic encryption. In FL systems incorporating the latter, user updates are encrypted before uploading to the server using public-private keys (Papernot et al. 2016). Moreover, since the source of the updates is not required for the aggregation, the updates can be transferred without including metadata related to the origin of the information. Lastly, to safeguard against data poisoning attacks, anomaly detection schemes may be employed on the encrypted updates to identify any outliers, and the nodes which contributed those malicious samples may be removed from subsequent rounds of updates. Further, a weight may also be assigned to each user update based on its quality, and this process may help in identifying clients which are helpful in faster convergence or higher performance of the model. Conversely, clients with lower ranked updates may be identified as stragglers.

To make the actual communication of updates over a network more resilient to eavesdroppers, the client may also consider sending the updates over a mixed network like Tor, or via a trusted third party (Ma et al. 2019).

6.5 Future Directions

The previous sections presented an outline of the concept, applications and issues related to the emerging area of Edge AI. It was mentioned that although appealing, the incorporation of distributed intelligence in the edge devices is not without its share of limitations which need to be addressed before Edge AI can be said to be mature. This section presents an overview of the future research avenues in the field of Edge AI.

6.5.1 *Open Issues in Federated Learning*

As mentioned in the previous section, convergence in FL systems is still not theoretically proven. More research efforts are required towards improving learning performance, that is bettering learning accuracy with lesser communication between the edge devices and the centralized server. The present tradeoff between privacy preservation mechanisms and convergence speed needs further investigation to tilt the balance in favor of faster training with maximal user privacy. Recognition and prevention of data and model poisoning attacks is still an open problem, as is the security of the transmitted updates against eavesdroppers. Lastly, the process of aggregation may be made robust by incorporating mechanisms like anomaly detection to identify outliers (malicious updates). The use of reward functions for participating nodes is still in infancy, and needs more study. Incorporation of rewards into the FL system would provide incentives to devices contributing more to the learning process (either due to their having more data, or more computational capability). Lastly, the use of Blockchain has also been proposed to facilitate secure transmission of updates (Kim et al. 2018). However, blockchain based federated learning systems have yet to become mainstream.

6.5.2 *Distributed Deep Reinforcement Learning*

Reinforcement learning, being the closest ML algorithm to human learning in the sense that it learns from experience, is another technique which can be explored for improving the intelligence in edge devices. Such distributed Deep Reinforcement Learning (DRL) (also referred to as multi-agent DRL) is expected to bring revolutionary improvements in the way interconnected edge devices learn and infer. This assumes particular importance in Edge AI scenarios where most sensors participate in data generation without being able to obtain or assign class labels. Semi-Supervised DRL has already been proposed for such cases (Mohammadi et al. 2017), and Unsupervised DRL for incorporating learning in Edge AI systems with little to no supervision is another open area of research.

6.6 Conclusion

This chapter first presented a discussion on the security threats to conventional edge computing systems. Thereafter, techniques to incorporate intelligence into the edge devices were highlighted. This is pertinent since Edge

AI is ultimately expected to allow and encourage collaboration between various edge nodes towards a globally intelligent model without explicit human support. An overview of the various threats to the rapidly growing field of Edge AI was then presented. Security issues in various aspects of Edge AI were discussed and some effective countermeasures were highlighted. Lastly, avenues for future research in the area were outlined wherein it was discussed that emerging technologies like Blockchain and Deep Reinforcement Learning could be leveraged to improve existing Edge AI systems.

References

Anley, Chris. 2002. *Advanced SQL Injection in SQL Server Applications.* Proceedings CGISecurity, 1–25.

Bagdasaryan, Eugene, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. 2018. How to Backdoor Federated Learning. *arXiv preprint arXiv:1807.00459.*

Biggio, Battista, and Fabio Roli. 2018. Wild Patterns: Ten Years after the Rise of Adversarial Machine Learning. *Pattern Recognition* 84: 317–331.

Bogdanoski, Mitko, Tomislav Suminoski, and Aleksandar Risteski. 2013. Analysis of the SYN Flood DoS Attack. *International Journal of Computer Network and Information Security (IJCNIS)* 5 (8): 1–11.

Brendel, Wieland, Jonas Rauber, and Matthias Bethge. 2017. Decision-based Adversarial Attacks: Reliable Attacks Against Black-Box Machine Learning Models. *arXiv preprint arXiv:1712.04248.*

Carlini, Nicholas, and David Wagner. 2017. *Adversarial Examples are Not Easily Detected: Bypassing Ten Detection Methods.* Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 3–14. ACM.

Carlini, Nicholas, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. *The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks.* 28th {USENIX} Security Symposium ({USENIX} Security 19), 267–284.

Chen, Weiteng, and Zhiyun Qian. 2018. *Off-Path {TCP} Exploit: How Wireless Routers Can Jeopardize Your Secrets.* 27th {USENIX} Security Symposium ({USENIX} Security 18), 1581–1598.

Chen, Eric Y., Yutong Pei, Shuo Chen, Yuan Tian, Robert Kotcher, and Patrick Tague. 2014. *Oauth Demystified for Mobile Application Developers.* Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, 892–903. ACM.

Chen, Pin-Yu, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. 2017a. *Zoo: Zeroth Order Optimization based Black-Box Attacks to Deep Neural Networks Without Training Substitute Models.* Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, 15–26. ACM.

Chen, Xinyun, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017b. Targeted Backdoor Attacks on Deep Learning Systems Using Data Poisoning. *arXiv preprint arXiv:1712.05526*.

Chen, Pin-Yu, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. 2018a. *EAD: Elastic-Net Attacks to Deep Neural Networks Via Adversarial Examples*. Thirty-Second AAAI Conference on Artificial Intelligence.

Chen, Yimin, Tao Li, Rui Zhang, Yanchao Zhang, and Terri Hedgpeth. 2018b. *Eyetell: Video-Assisted Touchscreen Keystroke Inference from Eye Movements*. 2018 IEEE Symposium on Security and Privacy (SP), 144–160. IEEE.

Cheng, Yu, Duo Wang, Pan Zhou, and Tao Zhang. 2017. A Survey of Model Compression and Acceleration for Deep Neural Networks. *arXiv preprint arXiv:1710.09282*.

Chollet, François. 2017. *Xception: Deep Learning with Depthwise Separable Convolutions*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1251–1258.

Cisco. 2016. Cisco Fog Director Cross-Site Scripting Vulnerability. Accessed 22 October 2019. https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20160201-fd.

Costin, Andrei. 2018. *IoT/Embedded vs. Security: Learn from the Past, Apply to the Present, Prepare for the Future*. Proceedings of Conference of Open Innovations Association FRUCT. FRUCT Oy.

Cretu, Gabriela F., Angelos Stavrou, Michael E. Locasto, Salvatore J. Stolfo, and Angelos D. Keromytis. 2008. *Casting Out Demons: Sanitizing Training Data for Anomaly Sensors*. 2008 IEEE Symposium on Security and Privacy (SP 2008), 81–95. IEEE.

Dalvi, Nilesh, Pedro Domingos, Sumit Sanghai, and Deepak Verma. 2004. *Adversarial Classification*. Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 99–108. ACM.

Damon, Evan, Julian Dale, Evaristo Laron, Jens Mache, Nathan Land, and Richard Weiss. 2012. *Hands-on Denial of Service Lab Exercises Using Slowloris and Rudy*. Proceedings of the 2012 Information Security Curriculum Development Conference, 21–29. ACM.

Das, Dipankar, Sasikanth Avancha, Dheevatsa Mudigere, Karthikeyan Vaidynathan, Srinivas Sridharan, Dhiraj Kalamkar, Bharat Kaul, and Pradeep Dubey. 2016. Distributed Deep Learning Using Synchronous Stochastic Gradient Descent. *arXiv preprint arXiv:1602.06709*.

Dean, Jeffrey, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato et al. 2012. Large Scale Distributed Deep Networks. *Advances in Neural Information Processing Systems*, 1223–1231.

Dhanapal, A., and P. Nithyanandam. 2017. *An Effective Mechanism to Regenerate HTTP Flooding DDoS Attack Using Real Time Data Set*. 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), 570–575. IEEE.

Dwork, Cynthia, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. *Calibrating Noise to Sensitivity in Private Data Analysis.* Theory of Cryptography Conference, 265–284. Berlin, Heidelberg: Springer.

Enev, Miro, Sidhant Gupta, Tadayoshi Kohno, and Shwetak N. Patel. 2011. *Televisions, Video Privacy, and Powerline Electromagnetic Interference.* Proceedings of the 18th ACM Conference on Computer and Communications Security, 537–550. ACM.

Fredrikson, Matt, Somesh Jha, and Thomas Ristenpart. 2015. *Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures.* Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 1322–1333. ACM.

Gao, Yansong, Chang Xu, Derui Wang, Shiping Chen, Damith C. Ranasinghe, and Surya Nepal. 2019. STRIP: A Defence Against Trojan Attacks on Deep Neural Networks. *arXiv preprint arXiv:1902.06531.*

Greenberg, Andy. 2017. The Reaper IoT Botnet has Already Infected a Million Networks. Accessed 13 January 2018. https://www.wired.com/story/reaper-iot-botnet-infected-million-networks/.

Hosseini, Hossein, Yize Chen, Sreeram Kannan, Baosen Zhang, and Radha Poovendran. 2017. Blocking Transferability of Adversarial Examples in Black-Box Learning Systems. *arXiv preprint arXiv:1703.04318.*

Howard, Andrew G., Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861.*

Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. 2016. SqueezeNet: AlexNet-Level Accuracy with 50x Fewer Parameters and <0.5 MB Model Size. *arXiv preprint arXiv:1602.07360.*

Ilyas, Andrew, Logan Engstrom, Anish Athalye, and Jessy Lin. 2018. Black-Box Adversarial Attacks with Limited Queries and Information. *arXiv preprint arXiv:1804.08598.*

Iorga, Michaela, Larry Feldman, Robert Barton, Michael J. Martin, Nedim S. Goren, and Charif Mahmoudi. 2018. *Fog Computing Conceptual Model.* No. Special Publication (NIST SP)-500-325.

Jagielski, Matthew, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. 2018. *Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning.* 2018 IEEE Symposium on Security and Privacy (SP), 19–35. IEEE.

Kim, Hyesung, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. 2018. On-Device Federated Learning Via Blockchain and Its Latency Analysis. *arXiv preprint arXiv:1808.03949.*

Konečný, Jakub, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated Learning: Strategies for Improving Communication Efficiency. *arXiv preprint arXiv:1610.05492.*

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 2012. Imagenet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems* 25: 1097–1105.

Kurakin, Alexey, Ian Goodfellow, and Samy Bengio. 2016. Adversarial Examples in the Physical World. *arXiv preprint arXiv:1607.02533*.

Ma, Chuan, Jun Li, Ming Ding, Howard Hao Yang, Feng Shu, Tony Q.S. Quek, and H. Vincent Poor. 2019. On Safeguarding Privacy and Security in the Framework of Federated Learning. *arXiv preprint arXiv:1909.06512*.

Madry, Aleksander, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv preprint arXiv:1706.06083*.

McIntosh, Michael, and Paula Austel. 2005. *XML Signature Element Wrapping Attacks and Countermeasures*. Proceedings of the 2005 Workshop on Secure Web Services, 20–27. ACM.

Mei, Shike, and Xiaojin Zhu. 2015. *Using Machine Teaching to Identify Optimal Training-Set Attacks on Machine Learners*. Twenty-Ninth AAAI Conference on Artificial Intelligence.

Mohammadi, Mehdi, Ala Al-Fuqaha, Mohsen Guizani, and Jun-Seok Oh. 2017. Semisupervised Deep Reinforcement Learning in Support of IoT and Smart City Services. *IEEE Internet of Things Journal* 5 (2): 624–635.

Moisejevs, Ilja. 2019. Evasion Attacks on Machine Learning (or "Adversarial Examples"). Accessed 22 October 2019. https://towardsdatascience.com/evasion-attacks-on-machine-learning-or-adversarial-examples-12f2283e06a1.

Na, Taesik, Jong Hwan Ko, and Saibal Mukhopadhyay. 2017. Cascade Adversarial Machine Learning Regularized with a Unified Embedding. *arXiv preprint arXiv:1708.02582*.

Nakhila, Omar, Afraa Attiah, Yier Jin, and Cliff Zou. 2015. *Parallel Active Dictionary Attack on wpa2-psk wi-fi Networks*. MILCOM 2015-2015 IEEE Military Communications Conference, 665–670. IEEE.

National Institute of Standards and Technology. 2010. National Vulnerability Database CVE-2010-3972 Detail. Accessed 22 October 2019. https://nvd.nist.gov/vuln/detail/CVE-2010-3972.

Nelson, Blaine, Marco Barreno, Fuching Jack Chi, Anthony D. Joseph, Benjamin I.P. Rubinstein, Udam Saini, Sutton Charles, J.D. Tygar, and Kai Xia. 2009. Misleading Learners: Co-opting Your Spam Filter. In *Machine Learning in Cyber Trust*, 17–51. Boston, MA: Springer.

Örs, Sıddıka Berna, Elisabeth Oswald, and Bart Preneel. 2003. *Power-Analysis Attacks on an FPGA–First Experimental Results*. International Workshop on Cryptographic Hardware and Embedded Systems, 35–50. Berlin, Heidelberg: Springer.

Papernot, Nicolas, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. 2016. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. *arXiv preprint arXiv:1610.05755*.

Paudice, Andrea, Luis Muñoz-González, Andras Gyorgy, and Emil C. Lupu. 2018. Detection of Adversarial Training Examples in Poisoning Attacks Through Anomaly Detection. *arXiv preprint arXiv:1802.03041*.

Roman, Rodrigo, Javier Lopez, and Masahiro Mambo. 2018. Mobile Edge Computing, Fog et al.: A Survey and Analysis of Security Threats and Challenges. *Future Generation Computer Systems* 78: 680–698.

Rosulek, Mike. 2017. *Improvements for Gate-Hiding Garbled Circuits*. International Conference on Cryptology in India, 325–345. Cham: Springer.

Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. *Mobilenetv2: Inverted Residuals and Linear Bottlenecks*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4510–4520.

Shi, Weisong, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal* 3 (5): 637–646.

Shokri, Reza, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. *Membership Inference Attacks Against Machine Learning Models*. 2017 IEEE Symposium on Security and Privacy (SP), 3–18. IEEE.

Sonar, Krushang, and Hardik Upadhyay. 2014. A Survey: DDOS Attack on Internet of Things. *International Journal of Engineering Research and Development* 10 (11): 58–63.

Steinhardt, Jacob, Pang Wei W. Koh, and Percy S. Liang. 2017. Certified Defenses for Data Poisoning Attacks. In *Neural Information Processing Systems Foundation, Inc*., Long Beach, 4–9, December 2017, 3517–3529, USA.

Suciu, Octavian, Radu Marginean, Yigitcan Kaya, Hal Daume III, and Tudor Dumitras. 2018. *When Does Machine Learning {FAIL}? Generalized Transferability for Evasion and Poisoning Attacks*. 27th {USENIX} Security Symposium ({USENIX} Security 18), 1299–1316.

Sun, San-Tsai, and Konstantin Beznosov. 2012. *The Devil is in the (Implementation) Details: An Empirical Analysis of OAuth SSO Systems*. Proceedings of the 2012 ACM Conference on Computer and Communications Security, 378–390. ACM.

Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing Properties of Neural Networks. *arXiv preprint arXiv:1312.6199*.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. *Going Deeper with Convolutions*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1–9.

Tramèr, Florian, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble Adversarial Training: Attacks and Defenses. *arXiv preprint arXiv:1705.07204*.

Truex, Stacey, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. 2019. Demystifying Membership Inference Attacks in Machine Learning as a Service. *IEEE Transactions on Services Computing*. https://ieeexplore.ieee.org/abstract/document/8634878

Udhayan, J., and R. Anitha. 2009. *Demystifying and Rate Limiting ICMP Hosted DoS/DDoS Flooding Attacks with Attack Productivity Analysis*. 2009 IEEE International Advance Computing Conference, 558–564.

Uesato, Jonathan, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. 2018. Adversarial Risk and the Dangers of Evaluating Against Weak Attacks. *arXiv preprint arXiv:1802.05666*.

Vanhoef, Mathy, and Frank Piessens. 2018. *Release the Kraken: New KRACKs in the 802.11 Standard*. Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 299–314. ACM.

Xiao, Yinhao, Yizhen Jia, Chunchi Liu, Xiuzhen Cheng, Jiguo Yu, and Weifeng Lv. 2019. Edge Computing Security: State of the Art and Challenges. *Proceedings of the IEEE* 107 (8): 1608–1631.

Xiaoming, Li, Valon Sejdini, and Hasan Chowdhury. 2010. Denial of Service (dos) Attack with UDP Flood. *School of Computer Science, University of Windsor, Canada*.

Zhang, Shanshan, Ce Zhang, Zhao You, Rong Zheng, and Bo Xu. 2013. *Asynchronous Stochastic Gradient Descent for DNN Training*. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 6660–6663. IEEE.

Zhou, Xiaoyong, Soteris Demetriou, Dongjing He, Muhammad Naveed, Xiaorui Pan, XiaoFeng Wang, Carl A. Gunter, and Klara Nahrstedt. 2013. *Identity, Location, Disease and More: Inferring Your Secrets from Android Public Resources*. Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, 1017–1028. ACM.