Defense against PUE Attacks in DSA Networks using GAN based Learning

Debashri Roy*, Tathagata Mukherjee†, Mainak Chatterjee*, Eduardo Pasiliao‡

* Computer Science, University of Central Florida, Orlando, FL 32826, {debashri, mainak}@cs.ucf.edu

† Computer Science, University of Alabama, Huntsville, AL 35899, {tm0130}@uah.edu

‡ Munitions Directorate, Air Force Research Laboratory, Eglin AFB, FL, 32542, {eduardo.pasiliao}@us.af.mil

Abstract—**Primary user emulation (PUE) attacks can pose a significant threat to the deployment of a robust cognitive radio network implementing dynamic spectrum access, for an intelligent allocation and usage of already crowded spectrum bands. In this paper, we present a solution towards the PUE attacks. We present two generative adversarial net (GAN) based models to successfully emulate the primary users (PUs) in two ways. We propose a (i) dumb generator model without any "prior" knowledge of PU's feature space, (ii) a smart generator model with some "prior" knowledge about PU's transmission. We also propose two deep neural network based discriminator models to discriminate between the PU and the emulated primary users (EPU) from the corresponding generators. Both the generator and discriminator of each GAN model gets smarter with iterative and sequential GAN training. Through a testbed evaluation, we show that discriminators are able to catch ∼50% of PUE attackers without the GAN training during the deployment phase. We also observe 100% accuracy for both the GAN models during training phase. Ultimately, after the GAN training, the discriminators achieved 98% and 99.5% accuracies, for dumb and smart generator models respectively, to distinguish "yet to be seen" PUE attacker.**

Keywords: PUE Attack, generative adversarial nets, deep neural network, software defined radios, confusion matrix.

I. Introduction

The opportunity for abundant usage of wireless devices has created an overly crowded radio spectrum and led to the scarcity in spectrum availability. In order to deal with this different pervasive measures are taken to deal with competitive nature of the spectrum availability. However, studies have shown that a large portion of licensed spectrum is unused at any given time or location [1]. To exploit such unused spectrum, the concept of dynamic spectrum access (DSA) was envisioned. The deployment of such spectrum management approach requires the use of intelligent systems at lower levels of the communication stack, even at the end users. Cognitive radio networks (CRN) have proven its competence for such deployments [2]. The basic idea of DSA is to allow some unlicensed users (secondary users) to opportunistically access the spectrum of the licensed users (primary users) when the spectrum is not in use. The rules strictly restrict any harmful overlap or pretentious use of spectrum by secondary users (SUs) when primary users (PUs) are present. The cognitive properties of CRN, enable the radio devices to take decision on how to manage spectrum for both PUs and SUs. One important challenge in order to achieve the goal of ideal CRN deployment is ensuring the security of PUs.

Most of the research on such spectrum allocation and management techniques for DSA deployment, are build on the assumption that all participants are cooperative, honest, and the network is without the presence of adversaries. The Federal Communications Commission (FCC) [3] mandated that all SUs must release the occupied spectral band as soon as any PU starts to transmit in that band, ensuring full privacy and availability for the licensed users. However, since till now the CRN network deployments are lacking any measure to implement the security guidelines [4], a situation could arise where the PUs get denied the required spectrum due to the presence of a malicious SU. This threat could be categorized as denial-of service (DoS) attack [5]. *An adversarial SU could pose itself as a PU by transmitting the signal with characteristics identical to the PU.* Such malicious SUs could threaten the integrity of the CRN in two ways: (i) by preempting the existing SUs in any spectral band, by posing as a PU; and (ii) by fooling the spectrum manager to deny the PU, as the malicious SU is impersonating itself as a valid PU. Such malicious deployment of large scale can hijack the entire white space of any spectrum band, thus launching a "DoS" attack on the legitimate SUs and PUs. Such attacks were first described by Chen et al. [6] as primary user emulation (PUE) attacks.

In this paper, we propose a machine learning based technique to detect PUE attacks. We design and implement a generative adversarial network (GAN) [7] based approach to generate malicious SUs. Any GAN based model works on a synergistic training of two neural networks: (i) a generator, and (ii) a discriminator. Therefore, we use generated malicious entities to train the discriminator for "yet to be seen" real PUE attackers. The main contributions of this paper are:

1) We impersonate SUs using two types of generator models: (a) dumb attacker, and (ii) smart attacker. The dumb attacker has no "prior" information about the signal characteristics of the PUs, but still tries to emulate the PUs. However, the smart attackers has sufficient information about the PU's signal data and therefore can emulate the PU's signal in an intelligent way.
2) We also model two discriminators (neural networks) and train them over the PU data, and generated data. The "dumb discriminator" gets generated data from dumb

generator attack model, and "smart discriminator" gets data from smart one. We show that the GAN training makes the discriminator able to distinguish between a wide array of possible malicious entity types and therefore being able to detect the real adversaries with intention of PUE attack.

3) Using USRP, we collect the raw over-the-air signal data from PUs to train both the discriminators. We also extract the PU's signal characteristics from the collected data and train the smart generator.
4) We show that the untrained discriminators have a $\sim$50% accuracy for detecting PUE attackers during the deployment phase.
5) We present 100% training accuracy for both the generator models. The trained discriminators over dumb, and smart generator models exhibit testing accuracy of 98%, and 99.5% respectively during deployment phase, with real PUE attackers present.

Next we present the background information and related works on PUE attacks.

II. Background and Related Works

In this section we discuss the main premise of PUE attack and its properties. We also present different ideas for defending against these attacks and finally we discuss the existing research on these concepts.

A. PUE Attack

PUE attack, first conceptualized and proposed by Chen et al. [6], is a DoS attack for CRNs. In a CRN, the PUs are prioritized over any SU. At the heart of a CRN, a spectrum manager makes the decision of preempting SUs when any PU is in need of transmission. For example, the PUs can be TV stations with a wide range, wireless microphones with a limited range [8], or mobile public safety devices surging with sudden transmissions during the times of emergency [9]. The SUs can be conceptualized as wireless devices connected to a WiFi network.

As per the policies mandated by FCC [3], SUs cannot cause any interference with PU's transmission, or hurt the PU's transmission in any other way. However, the malicious SUs can pretend to be a PU by "mimicking" certain features of PUs. One example is, where a malicious SU emulates itself to be a PU by using a low power commercial off-the-shelf TV transmitter [10] located near some legitimate PUs and starts to transmit on a particular spectrum band. The objective of the emulated primary user (EPU) could be of two types: (i) a selfish goal to maximize the spectrum usage for itself, (b) malicious goal with a tendency to prevent the legitimate SUs from detecting vacant spectrum bands, leading to a DoS attack. The selfish EPU attacker starts transmitting on a vacant frequency band without going into the waiting queue for SU selection by the central spectrum allocator. The malicious EPU attacker starts to "attack" over multiple vacant frequency bands randomly, resulting in starvation of PUs' and legitimate SUs.

B. Existing Defenses against PUE Attack

Any security threat can be thwarted by several defense mechanisms through continuous research. Though there is still a dearth of opportunities to come up with a robust defense mechanism to overcome all technical challenges related to PUE attacks, there are few existing defense mechanisms which are available today. One such technique is matched filtering-based detection [11] of PUE attacker, which requires specialized hardware and software. On the other hand, energy detection-based schemes [12] poises high risk of missed false alarm and missed detection possibility. The location-based detection technique [6] is limited to stationary PUs with known coordinates. Another approach was the use of phase noise as a signature [13] to identify PUs and defend against PUE attacks. The key idea behind this technique is to erase modulation from captured signal (which is modulated) and extract phase noise of local oscillator (LO) to work as a signature. This approach is also constrained over the prior knowledge of modulation scheme. However, the use of a cryptographic signature and wireless link signature [14] to detect PUE attacker requires an additional helper node in close physical proximity of each PU. Similarly, a cyclostationary calculation [15] based artificial neural network model needs prior knowledge of modulation schemes of PUs, and it is constrained to be different for the other users.

All the mentioned defense mechanisms are burdened with overheads or constraints of different types. To resolve this, we propose a GAN based robust PUE attack defense which will involve only centralized deployment and can provide defense mechanism for both mobile and stationary PUs or legitimate SUs, without any such constraints.

III. Proposed GAN Approach

In this section we propose a GAN based approach to present a robust defense mechanism for PUE attack. In a GAN, we have one generator and one discriminator. We propose two GAN based models: (i) one model which will work without any prior knowledge about the PU or the CRN, (ii) the other with the assumption, that the attacker will have prior information about the PUs characteristics for that CRN. So, for the first case we take the SU to work as a dumb emulated primary user (EPU). However, in the second case the EPU is smart to gather or know all the helpful prior information about the PU transmission, so we call those as smart EPUs. Few examples of prior knowledge that an attacker could leverage to mimic the PUs characteristics are: (i) a set of modulation schemes which the PUs use, (ii) the used frequency band, (iii) channel bandwidth for PU's transmission, (iv) geographical location of the PU's transmitter, (v) sample space and sample size of the PU transmission, and so on.

A. GAN based Problem Formulation for PUEA Defense

A GAN framework comprises of two distinct models: the generator ($\mathcal{G}$) which learns the real data distribution and generates the "mimicked" data, and the discriminator ($\mathcal{D}$) which tries to distinguish the mimicked data from the "real" data by estimating the probability that the sample came from the real data rather than the $\mathcal{G}$. The overall idea is that, during the training phase the generators will pose as a selfish or malicious PUE attacker and make the discriminator stronger in order to fight against the real selfish and malicious PUE attackers during the deployment phase. We call the selfish and malicious PUE attackers as emulated primary users (*EPU*), and the legitimate primary users as *PUs*. We use a centralized spectrum allocator and train it using the GAN model and therefore work as a robust discriminator during deployment.

a) Generative Model: The generative model has two main inputs, (i) prior information about the PU's feature space ($s(t)$), (ii) a additive Gaussian white noise ($n(t)$). The noise is added with the prior information, $z(t) = s(t) + n(t)$. The output $z(t)$ is then fed to the generator which works as a unsupervised learning tool and learns the data distribution ($p_z(z)$) of PU's feature space. Note that we have indexed the prior information, noise and the generated data by time t. This has been done to acknowledge the fact that the signal characteristics can change over time. However for this work we do not consider the effects of variation of the signal characteristics (prior information) and noise with time and essentially assume that $s(t) = s\ \forall t$ and $n(t) = n\ \forall t$. Hence we also get $z(t) = z\ \forall t$. Recall, there are two types of EPUs, one is dumb and another smart. The generator for dumb EPU is deprived of any prior information about the PU's feature space, so $s(t)$ for the generator model of dumb EPU is random and does not reflect the actual signal characteristics of the PU. The cost function of generator is denoted by $V(\mathcal{G})$. The GAN model's target is to minimize this cost, i.e., to minimize the probability of $\mathcal{D}$ correctly identifying the data from $\mathcal{G}$.

b) Discriminative Model: The discriminator is fed with both real data (x) drawn from a data distribution $p_{data}(x)$, and generated data from generator ($z(t)$). The objective of the discriminator is to successfully distinguish between these two types, i.e., to learn the difference between $p_{data}(x)$, and $p_z(z)$. The cost function for discriminator is denoted as $V(\mathcal{D})$. The target of discriminator is to maximize its cost, i.e., to increase probability to correctly identify samples from training examples and data from $\mathcal{G}$. Target of the overall GAN model is to minimize the cost for generator and maximize the cost for discriminator. The overall cost $V(\mathcal{G}, \mathcal{D})$ is formulated as $V(\mathcal{G}, \mathcal{D}) = \mathbb{E}_{p_{data}(x)} \log \mathcal{D}(x) + \mathbb{E}_{p_z(z)} \log(1 - \mathcal{D}(\mathcal{G}(z)))$, where $p_z(z)$ is the generator's distribution over generated data samples z, $p_{data}(x)$ is the data distribution over real data samples x, $\mathcal{D}(x)$ is the probability that x came from $p_{data}(x)$ than $p_z(z)$. $\mathcal{D}(\mathcal{G}(z))$ represents the probability that x came from $p_z(z)$ than $p_{data}(x)$. Objective of the GAN training is:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{G}, \mathcal{D}) \tag{1}$$

Throughout the training phase, the GAN framework eventually converges to an unique optimal discriminator for a particular generator, $\mathcal{D}^*(\mathbf{x}) = \dfrac{p_{data}(\mathbf{x})}{p_{data}(x) + p_z(z)}$. It is intuitive that $\mathcal{D}$ is optimal when the discriminator can distinguish between each real sample (x) and generated sample (z). Similarly, it is also deduced that $\mathcal{G}$ is optimal when the $\mathcal{D}$ cannot distinguish between x and z, $\mathcal{G}$ is optimal when $p_z(z) = p_{data}(x)$.

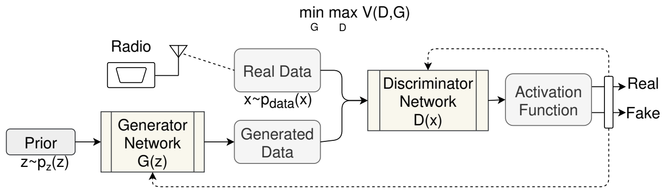


Fig. 1. Implementation of GAN in RF Domain

c) Proposed GAN Architecture: The overall GAN architecture is shown in Fig. 1. Once the generator ($\mathcal{G}(z)$) is trained, it generates "mimicked" data from the data distribution ($p_z(z)$) with the prior information. The trained discriminator knows the difference between the mimicked data distribution ($p_z(z)$) and real data distribution ($p_{data}(x)$), so it will try to distinguish the mimicked data (z) from the real data (x). The activation function at discriminator is *sigmoid*. The feedback from the output is fed to both $\mathcal{G}(z)$, and $\mathcal{D}(z)$. So ultimately the target of overall GAN model is to maximize the discriminator's cost and minimize the generators cost, mathematically formulated as equation (1).

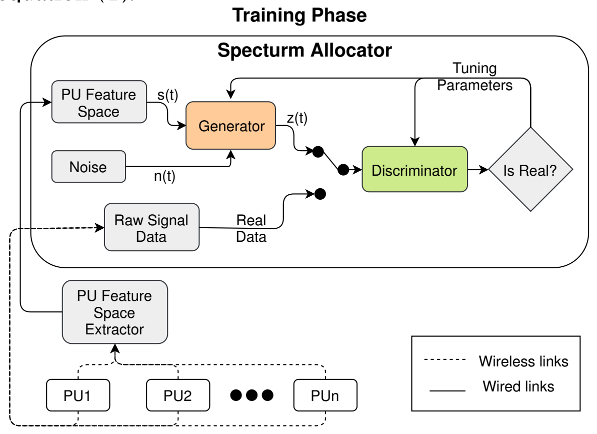


Fig. 2. GAN Training in the Spectrum Allocator

B. Proposed GAN based Approach for PUEA Defense

The proposed method is compatible with the existing system of centralized spectrum allocation for dynamic spectrum access implementations. The idea is to install a generator and discriminator model inside the spectrum allocator. During the training phase, the discriminator is trained over the generated data and real signal data from the legitimate PUs. The overall proposed training approach is presented in Fig. 2. We train the discriminator over the generator model. We consider the generator to be either one of the two types: (i) a generator posing as a dumb EPU, (ii) a generator posing as a smart EPU. The smart EPUs collect the prior information about the feature space of the PUs from a feature space extractor. The discriminator gets trained over both real (x) and generated data (z) over iterations and become robust enough to distinguish between the real and synthetic data. The output of the discriminator is fed to both the generator and discriminator models so as to tune-up the hyper-parameters of each model, depending on the result. In this way, the generator also gets smarter over each iteration and the discriminator gets smarter than the generator over the next iteration. Thus we design the training in such a way that over time the discriminator eventually over-powers the smartest possible generator model. Once the GAN training is complete, the trained discriminator is deployed in the spectrum allocator.

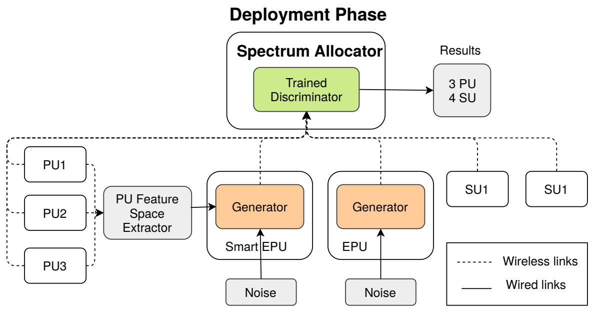


Fig. 3. Deployment of Trained Discriminator in Spectrum Sensing Scenario

During the deployment phase, the centralized spectrum allocator gets signal information from spectrum sensors, and pass them through the trained discriminator before the spectrum is allocated. The overall proposed approach for deploying the trained discriminator is shown in Fig. 3. In the figure we consider 3 legitimate PUs, 1 dumb EPU, 1 smart EPU, and 2 SUs. The dumb EPU does not have any prior knowledge about PU's feature space, but the smart one has. The trained discriminator is able to recognize the 2 EPUs as SU, despite the effort of the malicious entities.

IV. Implementation and Results

For implementing the GAN model, we use the proposed generator and discriminator models with data collected from over-the-air RF transmission. Next we describe the data collection, experimental setup, implementation details, and experimental results.

A. Used Dataset and Experimental Setup

We use a dataset of raw I/Q values from 8 software defined radios (universal software radio peripheral (USRP) B210 [16]). The details of I/Q data generation and data collection mechanism is similar to [17], and [18]. Each dataset comprises T training samples (for T timestamps) and a sample size of N, where each sample is a vector $(I, Q) \in \mathcal{C}$ representing a number in the complex plane. Each vector at timestamp t is represented as: $x(t) = [[(I, Q)_i]^t; i = 1, 2, \cdots, N] \in \mathcal{C}^N$ for timestamp $t = 1, 2, \cdots, T$. We use these vectors as input during GAN training. Note that though we collect the data with respect to different time stamps, we do not treat the data thus collected as a time series for this work, that is we *ignore* the temporal aspect of the data.

We conducted the testbed evaluation on a Ryzen 8 Core system with 64 GB RAM, a GTX 1080 Ti GPU unit, and 11 GB memory. We used different machine learning libraries to design the proposed GAN models. We useu *Keras* [19] as the frontend and *Tensorflow* [20] as the backend for the neural network architectures used in GAN. We also use *Numpy*, *Scipy*, and *Matplotlib* python libraries for implementation of different operations.

B. GAN Model for Dumb PUE Attacker

In this case, the generator has no prior information. The generator starts by randomly generating data within the sample space $[X, Y]$, where X and Y are random numbers. For our experiments, we took X as 0 and Y as 1. This dumb generator picks a random sample size N. The randomly generated data is then used as input to three *dense* layers of sizes $N/2$, N, and $2N$ respectively with *tanh* activation function in the first two layers. The third neuron layer uses *sigmoid* activation function, and generates data of size $2N$.

The generator $\mathcal{G}(z)$ continues to learn the data distribution ($p_z(z)$) and generates fake samples of size $2N$ within a random sample space. The discriminator ($\mathcal{D}(x)$) for the dumb generator is simple. We first have one input layer of $2N$ nodes, which is followed by two hidden layers of N and $N/2$ nodes respectively. Ultimately, a *softmax* output layer of 2 nodes is used to identify whether the input is from a legitimate SU or malicious EPU. We use *tanh* as activation function at the hidden layers and add *Dropout* [21] of 0.5 in between those layers to avoid overfitting. We also use $l1$ regularization for the same purpose. The output from the last layer is fed to the layers of generator as well as discriminator in the next epoch for tuning the parameters of both. One epoch is actually a combination of a forward and a backward pass through the designed model over the entire dataset. The overall GAN implementation using dumb generator is shown in Fig. 4. In this case, the $\mathcal{G}(z)$, and $\mathcal{D}(x)$ are respectively 3-layered and 5-layered networks. In the figure, the solid lines represent the connection between two layers, whereas the dotted lines represent the feedback from output layer of the discriminator to the other layers of both generator and discriminator to be used in next epoch.

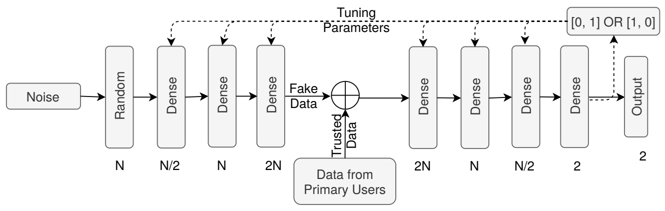


Fig. 4. GAN Implementation for Dumb Emulated Primary User

C. GAN Model for Smart PUE Attacker

In this case, generator $\mathcal{G}(z)$ has prior knowledge about some features of PUs. We get a sample of size 1024 and sample space of [-1, 1] from the PU feature extractor. These features eventually help the EPUs to better mimic legitimate PUs. In this set of experiments, we get the sample size (N) and the sample space from the "feature space extractor". Feature space extractor is a module in the spectrum allocator, which extracts features and builds a feature space for the PUs from the sensed signal data. The GAN implementation with a smart generator is shown in Fig. 5. For the sake of consistency and generality, we use the same number of layers as the dumb one, for modeling the generator of smart EPU. However, knowing that the generator is smart, we design the discriminator in a way that it is 1-layer more deep than the one for the dumb one. We use 2 *dense* layers with N nodes instead of 1. So now the $\mathcal{D}(x)$ becomes a 6-layered network. The rest of the generator and discriminator properties are similar as before.

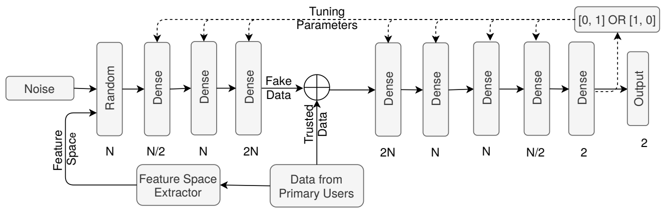
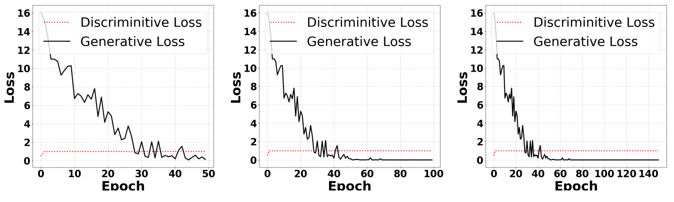


Fig. 5. GAN Implementation for Smart Emulated Primary User

D. Experimental Results

We perform two sets of experiments here: (i) first we train the GAN model on a dumb generator and use the discriminator as discussed in section IV-B, (ii) next we train our proposed model on the smart generator and use a discriminator as described in section IV-C. For both the cases we performed the training on the data collected from the authentic PUs. We notice that the discriminator trained on the dumb generator was able to detect the EPUs with 49.22% accuracy before the GAN training. Whereas, the discriminator trained on smart generator was able to do that with 50.15% accuracy. This slight improvement is due to the one extra layer of neurons in the smart discriminator. However, the overall achieved accuracy is far below expectations. The discriminator is naive and can distinguish the EPUs from the legitimate PUs only with $\sim 50\%$ of accuracy, which is similar to what one would obtain for random guessing.

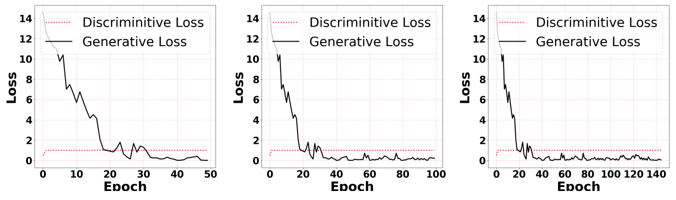
1) Training Phase: We train both the generator and discriminator through iterative sequential learning to strengthen the generative and discriminative model over time. We use *categorical cross-entropy* training on *Adam* [22] optimizer for *gradient based optimization*. We use 90%, 5%, and 5% of the total data for training, cross-validation and testing respectively.



(a) 50 Epochs (b) 100 Epochs (c) 150 Epochs

Fig. 6. Generator and Discriminator Loss of GAN Model with Dumb EPU

We experimented with several training paradigms for training the GAN models in order to get the best performance from it. This leads us to train both the models in a 3-step approach. We first train the models for 50 epochs with learning rate of 10^{-4} and 10^{-3} for generator and discriminator respectively. Next, we train the models for another 50 epochs with lower learning rates of 10^{-5} and 10^{-4} for $\mathcal{G}(z)$ and $\mathcal{D}(x)$ respectively. In the last 50 epochs, we decrement both the learning rates by $1/10$ again. Notice that in each epoch, the learning rate of generator is lower than the discriminator, as we want the generator to learn precisely about the possible data distribution $(p_z(z))$ of PUs for better EPU emulation. However, we give the discriminator more layers to be able to accurately learn the decision boundary in order to eventually overpower the generator. It is clear from the Fig. 6, that the generator's loss starts to decrease and the discriminator's loss starts to increase at the beginning. However, after a certain number of epochs, both the losses saturate. The generator-loss, with the dumb generator fluctuates more and reaches saturation later (after 30 epochs); whereas the generator-loss with smart generator reaches saturation within less than 20 epochs, as shown in Fig.7. This result bolsters the intuition that knowing some of the PUs features will make the generators smart at emulating the PUs and this in turn will make the system converge faster. The discriminator's behavior is more or less stable for both the models. Once both the models are trained, we proceed to the deployment phase.



(a) 50 Epochs (b) 100 Epochs (c) 150 Epochs

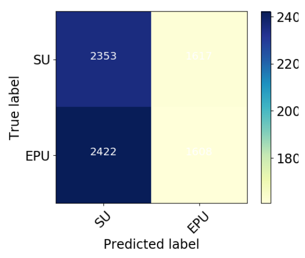
Fig. 7. Generator and Discriminator Loss of GAN Model with Smart EPU

2) Deployment Phase: During the deployment phase, we randomly choose over-the-air signal data from EPUs and SUs. We collect 4000 signal data from legitimate SUs and 4000 signal data from EPUs. We programmed USRP B210s [16] to work as SUs and EPUs. These EPUs use same sample size and sample space as the PUs, making themselves indistinguishable to the normal discriminator (trained without GAN). We observe $\sim 50\%$ EPU detection rate prior to GAN training. However, after the discriminator is trained and learned the data distribution $(p_z(z))$ of the EPUs, we get better detection rate. We achieved a 98.04% accuracy from the discriminator which was trained using the dumb generator and 99.5% accuracy from the discriminator which was trained using the smart generator. These results are presented in Table I.

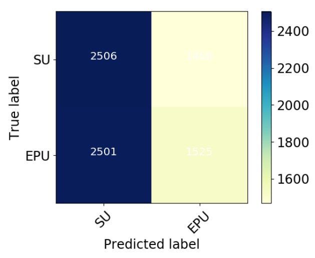
	Before GAN Training	After GAN Training
$\mathcal{D}$ with Dumb $\mathcal{G}$ Training	49.22%	98.04%
$\mathcal{D}$ with Smart $\mathcal{G}$ Training	50.15%	99.5%

TABLE I
ACCURACIES FOR DIFFERENT IMPLEMENTATIONS

3) Performance Analysis: Using accuracy as a measure of efficacy of an algorithm can sometimes be incomplete and misleading depending on the type of data, such as the case for skewed data. A confusion matrix overcomes those problems by showing a relative relation between false positives and false negatives and actual data labels. Hence we present the confusion matrices for both the experiments of deployment phase, in Fig. 8, and 9. It is evident from Fig. 8, that almost

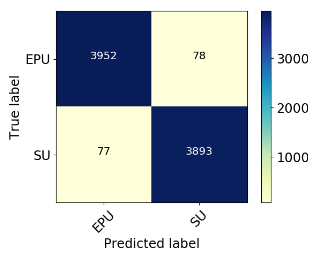


(a) Dumb Generator

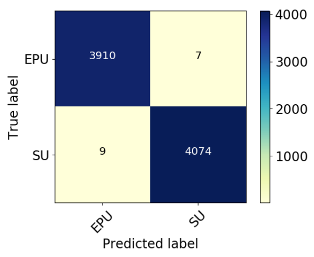


(b) Smart Generator

Fig. 8. Confusion Matrices with Dumb and Smart EPUs: before GAN Training



(a) Dumb Generator



(b) Smart Generator

Fig. 9. Confusion Matrices with Dumb and Smart EPUs at Deployment Phase: after GAN Training

all of the signals are predicted to be transmitted from SU, giving an accuracy of $\sim$50%. So, the discriminator is not able to distinguish between EPU and SU, thus predicting all as SU. However, the confusion matrices after the GAN training has lower false positive and false negative rates. We also notice that the false positives and false negatives are higher in case of the GAN model which was trained over dumb generator. In summary,

- We achieve $\sim$50% accuracy of EPU detection using the discriminator trained without the GAN for both type of proposed models, during deployment phase.
- During training, we observe 100% training accuracy for both dumb and smart GAN models.
- The trained discriminator with dumb and smart generator training gives testing accuracy of 98% and 99.5% respectively during the deployment phase.
- The proposed GAN based model can be applied for any type of cognitive radio transmitter irrespective of PU or SU's properties. We achieve 98-99% accuracy of EPU detection after the dumb generator training. So without any knowledge of PU properties, the proposed model is capable of detecting PUE attackers with 98% accuracy. Some "prior" information can boost up the accuracy to 99.5%.

V. Conclusions

In this paper we present a robust defense mechanism against PUE attack in a cognitive radio network. We design GAN based models considering (a) no prior information and (b) prior information about the PUs. We call them as dumb and smart GAN models respectively. Through testbed evaluations, we show that the GAN training for both kind of generators give a competitive accuracy for EPU or PUE attacker detection during the deployment phase. However, GAN model with smart generator training achieves better accuracies and faster saturation than the dumb one. Both models are able to detect the malicious and selfish PUE attacker with more than 98% of accuracy. Extending this concept towards providing security for other issues in wireless communication, could be one of the next steps to consider.

References

[1] M. McHenry, "Spectrum white space measurements," New America Foundation Broadband Forum, 2003.

[2] Q. Zhao and B. M. Sadler, "A Survey of Dynamic Spectrum Access," *IEEE Signal Processing Magazine*, vol. 24, no. 3, pp. 79–89, 2007.

[3] FCC ET Docket No. 08-260, "Second Report and Order and Memorandum Opinion and Order – Unlicensed Operation in the TV Broadcast Bands / Additional Spectrum for Unlicensed Devices Below 900 MHz and in the 3GHz Band," US Federal Communications Commission, Washington DC, FCC 08-260, 2008.

[4] S. Bhattacharjee, S. Sengupta, and M. Chatterjee, "Vulnerabilities in cognitive radio networks: A survey," *Elsevier Computer Communications*, vol. 36, no. 13, pp. 1387–1398, 2013.

[5] G. Jakimoski and K. P. Subbalakshmi, "Denial-of-Service Attacks on Dynamic Spectrum Access Networks," in *IEEE International Conference on Communications Workshops*, 2008, pp. 524–528.

[6] R. Chen, J. Park, and J. H. Reed, "Defense against Primary User Emulation Attacks in Cognitive Radio Networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 1, pp. 25–37, 2008.

[7] I. Goodfellow *et al.*, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[8] S. Chen, K. Zeng, and P. Mohapatra, "Hearing is believing: Detecting mobile primary user emulation attack in white space," in *Proceedings IEEE INFOCOM*, 2011, pp. 36–40.

[9] R. Chen, J. . Park, and K. Bian, "Robust Distributed Spectrum Sensing in Cognitive Radio Networks," in *Proceedings IEEE INFOCOM*, 2008, pp. 1876–1884.

[10] Lime Microsystems, "A Pocket Digital TV Transmitter with Raspberry Pi Zero and LimeSDR Mini," https://www.crowdsupply.com/lime-micro/limesdr-mini, 2019.

[11] D. Cabric, S. M. Mishra, and R. W. Brodersen, "Implementation Issues in Spectrum Sensing for Cognitive Radios," in *Asilomar Conference on Signals, Systems and Computers*, vol. 1, 2004, pp. 772–776.

[12] M. P. Olivieri, G. Barnett, A. Lackpour, and A. D. and, "A Scalable Dynamic Spectrum Allocation System with Interference Mitigation for Teams of Spectrally Agile Software Defined Radios," in *Proceedings IEEE DySPAN*, 2005, pp. 170–179.

[13] C. Zhao, W. Wang, L. Huang, and Y. Yao, "Anti-PUE Attack Base on the Transmitter Fingerprint Identification in Cognitive Radio," in *Proceedings IEEE WiCom*, 2009, pp. 1–5.

[14] Y. Liu, P. Ning, and H. Dai, "Authenticating Primary Users' Signals in Cognitive Radio Networks via Integrated Cryptographic and Wireless Link Signatures," in *Proceedings IEEE SP*, 2010, pp. 286–301.

[15] D. Pu, Y. Shi, A. V. Ilyashenko, and A. M. Wyglinski, "Detecting Primary User Emulation Attack in Cognitive Radio Networks," in *Proceedings IEEE GLOBECOM*, 2011, pp. 1–5.

[16] Ettus Research, "USRP B210," https://www.ettus.com/product/details/UB210-KIT, 2018.

[17] D. Roy, T. Mukherjee, M. Chatterjee, and E. Pasiliao, "Detection of Rogue RF Transmitters using Generative Adversarial Nets," in *Proceedings IEEE WCNC*, 2019.

[18] D. Roy, T. Mukherjee, and M. Chatterjee, "Machine Learning in Adversarial RF Environments," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 82–87, 2019.

[19] F. Chollet *et al.*, "Keras: The Python Deep Learning library," https://keras.io, 2015.

[20] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, 2016.

[21] N. Srivastava *et al.*, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[22] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2014.