Australian Government

Department of Defence

Science and Technology

Adversarial Machine Learning for Cyber-Security: NGTF Project Scoping Study

AMLC Team at UniMelb [1], *Data61* [2] *and Swinburne Univ.* [3]

Tamas Abraham [4], *Olivier de Vel* [5], *Paul Montague* [4]

[1] **School of Computing and Information Systems, University of Melbourne**

[2] **Data61, CSIRO**

[3] **School of Software and Electrical Engineering, Swinburne University**

[4] **Cyber Assurance and Operations, Cyber & Electronic Warfare Division (CEWD)**

[5] **Principal Scientist (Cyber), CEWD**

Defence Science and Technology Group

DST-Group–GD–0988

ABSTRACT

This report is the result of a scoping study undertaken as part of an Australian Department of Defence Next Generation Technologies Fund (NGTF) project entitled *Adversarial Machine Learning for Cyber-Security* (AMLC). The report describes the broader context for the project (e.g. attacks and defences against machine learners), outlines general concepts and techniques for adversarial machine learning, and focusses on reinforcement machine learning algorithms of specific relevance to Defence. A software simulation platform will also be developed to demonstrate the effectiveness of the attacks against, and defences of, such machine learning algorithms in a cyber-security context.

Produced by

Cyber and Electronic Warfare Division
PO Box 1500
Edinburgh, South Australia 5111, Australia

Telephone: 1300 333 362

January, 2018
AR-017-073

APPROVED FOR PUBLIC RELEASE

Adversarial Machine Learning for Cyber-Security:

NGTF Project Scoping Study

Executive Summary

Impactful applications of machine learning (ML) in Defence abound, from cyber-security (e.g. network security operations, malware analysis) to machine reasoning and autonomous systems (e.g. decision-making and platform control systems, computer vision, speech recognition, speaker identification etc.). But despite the many successes, the very property that makes machine learning desirable—adaptability—is a vulnerability able to be exploited by an economic competitor or state-sponsored attacker that could potentially result in the severe degradation of the integrity, security and performance of Defence systems. Attackers aware of the ML techniques being deployed against them can, for example, contaminate the training data to manipulate a learned ML classifier in order to evade subsequent classification, or can manipulate the specific metadata upon which the ML algorithms make their decisions and exploit identified weaknesses in these algorithms so called Adversarial Machine Learning (AML).* The resilience of learning algorithms is thus a critical component for trustworthy systems in Defence, National Security and society more broadly, but one that is so far poorly understood.

This report is a scoping study undertaken in the context of an Australian Department of Defence Next Generation Technologies Fund (NGTF) project entitled *Adversarial Machine Learning for Cyber-security (AMLC)*.† The project will investigate concepts, techniques and technologies relating to the security of machine learning algorithms and machine learning systems. The project will focus on a specific class of ML algorithms, namely, reinforcement learning (RL) algorithms that are used in an increasing number of Defence-relevant application settings such as decision-making systems, and in cyber-security settings such autonomous cyber-security operations (ACO), malware detection etc.

The NGTF AMLC project scoping study aims to:

(i) identify novel, deep research problems in adversarial machine learning for cyber-security

(ii) ground the research programme design in the existing AML and cyber-security literature, so that future work builds on a firm foundation of existing knowledge

(iii) develop relevant AML techniques and technology solutions

(iv) ground the AMLC research in practical problems of interest to DST Group and its Defence and

*Adversarial machine learning is also sometimes referred to as 'machine learning security', or interpreted more broadly as 'end-to-end machine learning security' or even 'security of system-of-machine-learning-systems' etc. We, in this project, will focus more specifically on the security of machine learning algorithms *per se*. We do not consider vulnerabilities in the machine learning software code.

†The AMLC project is supported by Australian Department of Defence NGTF funds and contribution of personnel from CEWD-DST Group and Data61.

National Security clients, through a flexible demonstration platform

(v) detail a plan of work packages, to ensure appropriate long-term project outcomes.

In particular, the AMLC project seeks to deliver:

- one or more practical platforms for demonstrating the various adversarial machine learning capabilities
- actionable knowledge of how sophisticated adversaries can exert unwanted influence on machine learning systems
- new robust machine learners built for adversarial environments.

The AMLC research team consists of members from Cyber and Electronic Warfare Division (CEWD) DST Group, Data61, the University of Melbourne, and Swinburne University.

Contents

Figures

1. Introduction

Vulnerabilities in machine learning are neither bogeyman nor science fiction: learning attacks on core cyber-security infrastructure have been recorded in the wild (see inset and example scenarios in Section 2).

Case study: In 2012 Google acquired 8 year-old anti-virus aggregator VirusTotal, an online service that runs and aggregates several dozen A/V scanners used by the majors (Symantec, Kaspersky, F-secure, etc.) for sharing threat intelligence. Two years later independent researcher Brandon Dixon uncovered three hacker groups (two nation-state backed) using VirusTotal to refine their malware—Comment Crew tied to China's military, NetTraveler in China, and a group believed to be in Iran. What Dixon found in VirusTotal logs, were traces of successful adversarial learning evasion attacks unfolding against every available commercial scanner. He observed lines of code repeatedly added, deleted and modified until the malware could go undetected [115].

Experts from across academia, industry and governments view robust learning systems as a pressing issue. The influential 2010 paper [94] by Sommer (ICSI) & Paxson (UC Berkeley) with over 500 citations in the flagship *IEEE Symposium on Security & Privacy* summarised challenges to using machine learning in network intrusion detection, as: data scarcity, poor scientific methodology, and vulnerability of machine learning in the face of adversaries. Adversarial machine learning (AML) seeks solutions to this third challenge, while mitigating the other two.

The 2015 research priorities statement by Russell (UC Berkeley), Dewey (Oxford) and Tegmark (MIT), short lists 'security for robustness' as vital for enjoying future benefits of machine learning [89], placing adversarial machine learning under the same umbrella as safety of self-driving cars. While in October 2017, aligned with its mission to enable breakthrough technologies for U.S. national security, DARPA hosted a 'Safe ML' workshop at the Berkeley Simons Institute led by Turing Awardee Shafi Goldwasser (MIT), bringing together researchers in adversarial learning and related areas, who concluded that there are many deep and impactful challenges facing the area. McAfee recently listed adversarial machine learning as the top threat in its 2018 forecast [15].

Case study: In a second incidence of an adversarial machine learning attack leveraging VirusTotal (a mistraining attack), Russian anti-virus firm Kaspersky Lab is believed to have submitted benign, system-critical files to VirusTotal, labelled as malicious. As Kaspersky's rivals trained their scanners on the new samples, those scanners began flagging legitimate system files as malicious, with serious consequences [60].

While much of adversarial learning to date has presented in-principle results, this project aims to demonstrate the consequences of learning attacks and effective counter measures on real-world systems.

Specifically we seek decision making for autonomous defence of computer networks, able to 'fight through' a contested environment—in particular adversarial machine learning attacks—and ensure critical command and control services (e.g. email servers, file servers, etc.) are preserved as much as possible.[1] This project will focus on adversarial machine learning concepts, techniques and technologies specifically aimed at reinforcement learning (RL) for cyber-security.[2] There is a range of applications of RL to cyber-security, including

[1] Human interaction is not precluded in our autonomous decision making scenarios. Human-in-the-loop operators may provide feedback to the system for training, or act as resources within the decision making itself.

[2] Reinforcement learning is a general-purpose framework for on-line decision-making. It is a class of 'machine learning

autonomous cyber operations, malware detection analytics, computer network operations etc. For example, one might consider a computer network in which each node has a set of possible observable states such as 'compromised' or 'vulnerable' or 'patched' or 'isolated', state transitions that reflect the spread of some attack stochastically into and then through the network, with actions selected by a strategic decision-maker in response. These actions could include 'patch', 'isolate', 'NOP', 'migrate' etc. Rewards for maintaining critical services and costs incurred when shutting down non-critical services, when optimised by these agents, ensure decisions align with the preferences of a human operator. There are clearly costs associated with various state, action pairs—with the cost varying for different nodes (e.g. critical services being lost). A reference for this work is [18].

To make use of a developed capacity for autonomous security operations, our demonstration platform will be built around software-defined networking (SDN), a next-generation tool chain for centralising and abstracting control of reconfigurable networks. SDN makes it possible to provide practical exhibition of our aims and results, providing multiple benefits, including

- as an emerging technology, it will become more relevant in a variety of application areas
- it provides a separation of logic from domain which permits robust decision making for other suitable applications
- as a platform, it will enable red and blue teaming of developed adversarial machine learning techniques
- its flexible centralised control makes it easier to develop demonstration scenarios
- a variety of complementary tools exist that make visual presentation of SDN demonstrations accessible to non-researchers.

Possible refinements could include introducing traffic types (e.g. classes of network protocols) into the edges (connections) between nodes. A network traffic classifier or anomaly detector could also effectively tackle the sub-problem of characterising the network traffic state. Communication could be flagged as anomalous for certain protocols and nodes inferred to be compromised. Actions of an RL algorithm given this information, could then include partial isolation in terms of blocking certain protocols between nodes. In reference to the concept of a mission as an overall goal, this could be introduced in the example as a certain scenario that needs to be supported e.g. certain services are required from certain nodes that must be protected (high cost of isolating them) if the mission is to succeed. There would still be a cost to shutting down non-critical services, as otherwise the RL algorithm would optimally shut everything non-critical down, potentially over-reacting to a small-scale attack. We should then learn a policy of action that respects the mission in the face of attack whilst, at the same time, being mindful of the possibility of state space explosion.

1.1. Project Aims

The *Adversarial Machine Learning for Cyber Project* seeks solutions to three significant grand challenges:

Aim 1: Use situation awareness from network analytics in a platform for strategic decision making in defence of mission-critical operational networks in adversarial environments.

from interaction' characteristic of autonomous software agents interacting in an environment. Agents ought to take *actions* in the environment so as to maximise some notion of cumulative *reward* over a time horizon. RL algorithms seek to learn a *policy* (a mapping from *states* to actions) that maximise the reward over time.

Aim 2: Evaluate machine learning systems for fundamental vulnerability to attacks through data manipulation.

Aim 3: Develop counter-measures to machine learning attacks, from filter approaches for strengthening existing adaptive systems, to principles for resilience-by-design.

2. Example Adversarial Machine Learning Scenarios

Adversarial machine learning is relevant wherever incentive exists to manipulate adaptive systems. The following scenarios illustrate that the challenges in making machine learning resilient arise in everyday applications. The questions raised permeate through numerous domains and remain relevant beyond any one specific example.

Figure 1: Can physical objects be manipulated (by 'sticker' or 'graffiti' attacks) to influence image recognition systems? (Image credit: Sturmovik at English Wikipedia CC3.0) [111].

Computer vision for self-driving cars.

While deep learning is accelerating state-of-the-art progress of object recognition in images, understanding why deep neural network models are successful and where their limitations might lie lag behind empirical success. Can real-world scenery—captured by digital imaging then fed to deep learning models—be manipulated to influence machine learning outputs (see Figure 1)? Preliminary demonstrations on images captured by mobile phones of benchmark dataset images [57] and of traffic Stop signs captured by moving vehicles suggest that blind spots demonstrated *in silico* exist *in vivo* also (e.g. misclassifying a Stop sign as a Speed Limit 45 sign using a printed 'sticker' perturbation [37]).

Naval fire control.

In combat situations, decision-making on a naval vessel must be timely, well coordinated, and leverage situation awareness despite partial observability, inherent uncertainty, and the risk of adversarial manipulation. Abstraction, detection, prediction and action in such a setting is similar to security operations in network security.

Malware detection in mobile security.

A front-line defence in mobile security is application (app) store malware scanning. So-called 'walled gardens' are well-known to guarantee a level of vetting of apps for exploits. However just as in consumer anti-virus (A/V) products (see insets on VirusTotal), app store scanners depend on machine learning to generalise from known examples to novel malicious code, and so are susceptible to learning attacks. Such attacks have been reported in the Google Play store for Android apps [81].

Security operations in software-defined networks.

Software-defined networks (SDN) is a next-generation technology that centralises the logical control of network management, ideal for the (semi-)automation of security operations (SecOps). While such automation permits fast, data-driven response to dynamic conditions, can sophisticated attackers exploit the adaptive nature of automated operations?

IoT security for MedTech devices.

While advanced electronics and even Internet-connectivity are bringing personalised medicine to medical technology (MedTech) devices, concern is growing that life-critical devices such as pacemakers and insulin pumps [40] are likely hackable.

While provably-secure approaches from formal methods may significantly reduce the MedTech device attack surface, products that control treatments based on historical data (such as continuous glucose monitors for diabetics, see Figure 2) may be susceptible to much more stealthy learning attacks. How can these systems be designed to be secure and robust?

Figure 2: How safe are MedTech devices such as continuous glucose monitors from learning attacks? (Image credit: Intel Free Press CC BY-SA 2.0) [3].

Utility grids.

The electricity grid has long been identified as a potential target for terrorist attack, in-part inspiring research into security of SCADA control systems [52]. With forecasting of demand factored into supply and general network control, adversarial machine learning is needed as a component of a robust utility grid.

Fraud detection in finance.

Machine learning drives numerous new technologies and applications in the finance industry (also under the porte-manteau FinTech) such as credit risk assessment, loan applications, and automated high-frequency trading. Cyber criminals clearly have incentives to manipulate fraud detection algorithms to circumvent financial safeguards.

3. Expected Project Deliverables

Following the project plan (refer to Section 6), the *Adversarial Machine Learning for Cyber Project* work packages (refer to Section 5) are designed to achieve the grand challenge aims stated in Section 1.1, to produce the following key deliverables:

Deliverable 1. Summary of current capabilities and trends in the adversarial machine learning literature.

Deliverable 2. A platform demonstrating newly-developed capabilities in adversarial machine learning, suitable for presentation to partners/clients in Defence and Data61/CSIRO.

Deliverable 3. Characterisation of costs in adversarial machine learning, across several domains.

Deliverable 4. New fundamental results on the limits of learning in adversarial environments.

Deliverable 5. New practical counter-measures for robust learning.

Deliverable 6. Internal reports and presentations communicating project topics and learnings to Defence and Data61/CSIRO.

Deliverable 7. Publications in leading international conferences/journals with team partners in Defence and Data61/CSIRO.

4. Background—Literature Review

This section starts with background on reinforcement learning, the primary area within machine learning relevant to the AMLC project, and software-defined networking, the technology behind our platform demonstrator. We follow this by a summary of newly discovered vulnerabilities of machine learning systems, and known countermeasures.

4.1. Strategic Decision Making and Software-Defined Networking

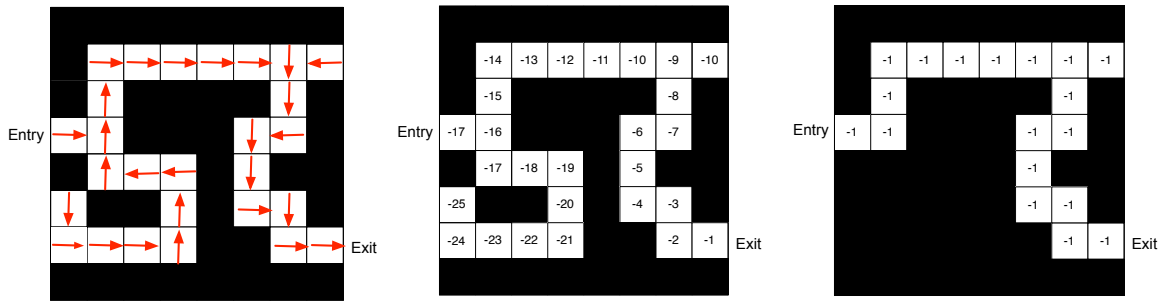
A key deliverable of the AMLC project is a demonstration decision-making platform for (semi)autonomous cyber operations. As discussed, a natural foundation for such a platform is *software-defined networking*—owing to its flexible reconfiguration and centralised control—complemented by *reinforcement learning*—an effective machine learning approach to decision making in dynamic and partially-observable environments. We begin our background review with overviews of both technologies.

4.1.1. Fundamentals of Reinforcement Learning

Inspired by the psychological and neuro-scientific models of natural learning, reinforcement learning is an adaptive machine learning method that is well suited to threat-aware response systems. Reinforcement learning is key in the ability to making reliable decisions under uncertainty that is inherent in autonomy and intelligence gathering. Recent years have witnessed a surge of success for reinforcement learning in a variety of domains. In addition to autonomous control of robots and helicopters: AlphaGo developed by Google acquisition DeepMind, has beaten multiple world champions at the board game Go; AlphaZero has recently been reported to "achieve *tabula rasa*[3] superhuman level performance" in the games of chess and Japanese shogi; Tesla's (partially) self-driving cars have been commercialised; and reinforcement learning has been used in economics, power systems, and more.

In all of the above examples, there is an *Agent*—e.g. the player in the game, the automobile-driver—that repeatedly observes the state of the environment, and takes actions accordingly in order to reach a certain goal. However, in RL the agent need not receive direct instructions about what actions to take, or what the consequences of those actions will be. These features distinguish RL from other learning problems, e.g. unlike supervised learning methods, where a model is learned on a fixed set of passively

[3]without using any domain knowledge except the rules of the game.



(i) Policy. *(ii) Value function.* *(iii) Model of the environment.*

Figure 3: Elements of reinforcement learning systems, depicted for a basic navigation problem.

observed independent and identically distributed (i.i.d.) training samples. In RL these samples are gathered throughout the training process in an active sequential manner.

Looking beyond the agent and environment, a reinforcement learning problem often includes four main elements [99]:

- **Policy.** A RL policy is a mapping from the observed states to the actions to be taken. In other words, it informs the agent what to do next. For instance, in the maze example [93] shown in Figure 3i), the arrows represent an example of a policy that indicates the direction of the next move. Note that the policy can be either deterministic or stochastic.

- **Reward signal.** The reward signal provides an immediate feedback on whether the action taken is good or not, with the agent's objective to maximise the cumulative rewards (possibly future-discounted) over the long run. It should be pointed out that although the agent can change the rewards they receive by taking different actions, they cannot alter the function that generates the reward signal.

- **Value function.** The value function predicts the future reward of a state, i.e. compared with the short term reward signal, it is the long term reward that can be accumulated over the future. Consider again the maze example: Figure 3ii) depicts the value function for each state.

- **Model of the environment.** A model of the environment aims to learn nature's behaviour, and uses this knowledge for improved planning. Normally there are two types of models—transition model and reward model, where the former predicts what the next state will be, and the latter predicts immediate reward. In the example of Figure 3iii), we illustrate the model that an agent might learn after observing a trajectory. Suppose that the agent has observed a trajectory through the maze that successfully reaches the goal. Then it builds a corresponding model of the environment. Note that the resulting model may not be accurate or even true.

Based on whether the agent represents the above elements, reinforcement learning can be categorised into the following types:

- **Value based.** Value-based algorithms maintain the value function, and select actions greedily by maximising value.

- **Policy based.** Instead of storing values for each state, policy-based algorithms only maintain a data structure for representing policy, e.g. the arrows in Figure 3i).

- **Actor critic.** Actor critic is a combination of the above two approaches, where the agent stores both the policy and the value function.

Another approach for classifying reinforcement learning agents is based on whether they explicitly build a representation of the environment first. Specifically, model-free agents directly learn the policy and/or the value function, while model-based agents first build up a model of how the environment works.

Planning in Reinforcement Learning. Model-based RL agents update policy and/or value function by randomly sampling simulated experience from the model, i.e. planning. Different techniques have been designed to accelerate this process. For example, the *Dyna* framework [97, 98] simultaneously uses the acquired (real) experience to build and update a model of the environment, and adjusts the value function (or the policy) using both simulated experience from the model and real experience from the environment. Since it takes significantly longer to gather real experience, the *Dyna* architecture is computationally more efficient than model-free learning. Another widely used technique is *prioritised-sweeping* [73]. Instead of randomly generating state-action during planning, it favours backups (i.e. updating a state value) that are likely to cause large value changes. Because only the state-action pair that leads to the terminal state has a positive value (e.g. the bottom-rightmost arrow in Figure 3i), it is less useful to backup along other pairs that have zero value.

4.1.2. Software-Defined Networking

In conventional online applications, the majority of the communication occurs directly between the client and the server. However, due to the wide use of content-delivery networks (CDN) and cloud services, it has become common for modern applications to access multiple servers and databases before data is finally returned to the client, resulting in a significantly higher proportion of “east-west" traffic. In addition, users nowadays may access an application with any type of device, e.g. smartphones, tablets and laptops, from anywhere around the world. All of these changes have produced ever-increasing strain on traditional networks, largely due to conventional switches having their own control unit, making network reconfiguration time consuming.

In order to better serve today’s dynamic and high-bandwidth applications, a new architecture called Software-Defined Networking (SDN) has emerged [2]. In SDN, we can program controllers, in the control plane, to define the data forwarding rules for the switches in the data plane. As can be seen from Figure 4 [41], there are three layers in the SDN architecture: (i) The application layer includes applications that deliver services. These applications communicate their network requirements to the controller via northbound APIs; (ii) The SDN controller translates these requirements into low-level controls, and sends them through southbound APIs to the infrastructure layer; (iii) The infrastructure layer comprises network switches that control forwarding and data processing.

One major advantage of SDN is that it decouples network control and forwarding functions, which makes the controller directly programmable. As a result, network resources can be conveniently managed, configured and optimised using standardised protocols. For example, Openflow [1] is a commonly-used protocol between the controller and the underlying switches. Note that in the simplest

implementation, the controller follows a centralised design where it has a global view of the entire network. However, in order to manage large-scale networks, hierarchical and distributed designs can also be adopted.

There have been a number of proprietary and open-source SDN controller software platforms, such as Cisco's Open SDN Controller [5], Floodlight [6], NOX/POX [8] and Open vSwitch [4]. In this project we have opted to use OpenDaylight [68, 9], which is the largest open-source SDN controller, and is updated regularly.

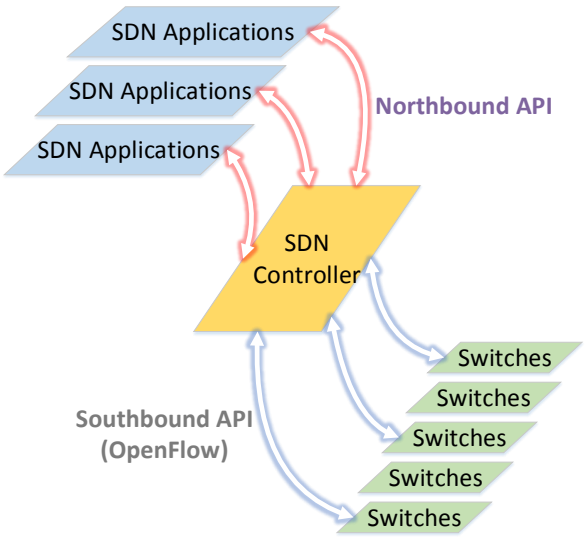


Figure 4: Software-defined networking architecture (adapted from [41]).

4.1.3. Applying RL in SDN

One main challenge that SDN faces arises from highly-dynamic traffic patterns, which motivate a requirement for the network to be reconfigured frequently. It has been demonstrated that RL is an ideal tool to accomplish such a task [90, 67, 65, 48, 109, 91, 70, 36, 55, 29]. For example, Salahuddin et al. [90] propose a roadside unit (RSU) cloud to enhance vehicle traffic flow and road safety. This RSU cloud is implemented using SDN, and leverages reinforcement learning to better cope with the dynamic service demands from the transient vehicles. In this way. the reconfiguration overhead can be minimised over the long run: increasing/decreasing the number of service centres, migrating services from one centre to another, for example. Mao et al. [67] design a job scheduling algorithm, called DeepRM, for computing clusters which have similar demands as SDN management. DeepRM uses colour images to represent system resources, e.g. CPU, RAM, as well as the resources required by each job slot, and exploits RL policy gradient methods to minimise average job slowdown.

Demonstrations of the suitability of reinforcement learning for SDN are still preliminary, at the basic proof of concept stage. Little has been shown of the RL for cyber-security, which poses challenges around large action spaces (requiring function approximation for value and policy functions), partial observability, and the presence of adversaries. We intend to investigate different potential candidate RL algorithms in the SDN context, such as Q-learning, SARSA, and Policy gradient methods, and also investigate attacks and counter-attacks against these algorithms.

4.2. Attacks on Machine Learners

We now focus on the ways in which attackers can target machine learning systems. This subsection first presents a taxonomy on attacks against (primarily) machine learning classifiers, and then summarises recent work that applies/modifies these attacks to manipulate RL systems.

4.2.1. Taxonomy of Attacks against Machine Learning Classifiers

Barreno et al. [17] develop a qualitative taxonomy of attacks against machine learning classifiers based on three axes: *influence*, *security violation* and *specificity*.

- The axis of *influence* concerns the attacker's capabilities: in *Causative attacks*, the adversary can modify the training data to manipulate the learned model; in *Exploratory attacks*, the attacker does not poison the training data, but carefully alters target test instances to flip classifications (e.g. by modifying one or more class labels). The resulting malicious instances which resemble legitimate data are called *Adversarial samples*.
- The axis of *security violation* indicates the consequence desired by the attacker: Integrity attacks are an example of deception attacks (achieving uncertainty, incompleteness, etc.), resulting in indecision, delayed decisions, poor or even wrong decisions, in decision-making systems. In this type of attack, the malicious instances bypass the filter as false negatives, whereas Availability attacks seek to cause a denial-of-service using legitimate instances.
- The axis of *specificity* refers to the target of the attacks. Indiscriminate attacks aim to degrade the classifier's performance overall, while Targeted attacks focus on a specific type of instance, or a specific instance.

Table 1 uses the taxonomy to classify previous published work on adversarial machine learning against supervised machine learners. As can be seen, most of the focus has been paid to exploratory integrity attacks. Presently, the Fast Gradient Sign Method (FGSM) attack [43] is the most widely studied method, and the C&W attack [31] is the most effective found so far on the application domains tested, mostly in computer vision. Both of these attack methods can be used for targeted or indiscriminate attacks. As the taxonomy was designed for supervised machine learners, we include attacks on reinforcement learning in a separate section (see Section 4.2.2).

Further examining the attacker's capabilities, in addition to potential control over the training data, a powerful attacker may also know the internal architecture and parameters of the classifier. Therefore, a fourth dimension can be added to the above taxonomy according to attacker information: in *White-Box Attacks*, the adversary generates malicious instances against the target classifier directly; while in *Black-Box Attacks*, since the attacker does not possess full knowledge about the model, he/she first approximates the target's model by training over a dataset from a mixture of samples obtained by observing the target's performance, and synthetically generates inputs and label pairs. Then if the reconstructed model generalises well, the crafted adversarial examples against this model can be transferred to the target network and induce misclassifications. Papernot et al. [85, 84] have demonstrated the effectiveness of the black-box attack in some limited domains. Specifically, they investigate intra- and cross-technique transferability between deep neural networks (DNNs), logistic regression, support vector machines (SVMs), decision trees and the k-nearest neighbour algorithm.

4.2.2. Attacks against Reinforcement Learning-based systems

In more recent studies, a small number of papers have begun to study whether attacks against classifiers can also be applied to RL systems. Huang et al. [49] show that deep reinforcement learning is vulnerable to adversarial samples generated by the Fast Gradient Sign Method [43]. Their experimental results demonstrate that both white-box and black-box attacks can be effective, even though the less knowledge the adversary has, the less effective the adversarial samples are.

Behzadan & Munir [20] establish that adversaries can interfere with the training process of Deep Q-Networks (DQNs), preventing the victim agent from learning the correct policy. Specifically, the attacker applies minimum perturbation to the state observed by the target, so that a different action is chosen as the optimal action at the next state. The perturbation is generated using the same techniques proposed against DNN classifiers. In addition, the authors also demonstrate the possibility of policy manipulation, where the victim ends up with choosing the actions selected by the adversarial policy.

Lin et al. [66] propose two kinds of attacks against deep reinforcement learning agents. In strategically-timed attacks, instead of crafting the state at each time step, the adversary identifies a subset of most vulnerable steps, and uses the C&W attack [31] to perturb the corresponding states. In *enchanting attacks*, the adversary uses sampling to iteratively find a sequence of actions that will take the agent to the target state, and craft the current state so that the agent will follow the next required action.

4.3. Adversarial Machine Learning Defences

A number of counter measures have been proposed since the discovery of adversarial samples. These can be roughly categorised into two classes: *data-driven defences* and *learner robustification*.

4.3.1. Data-driven Defences

The first class of defences are data driven—they either filter out the malicious data, inject adversarial samples into the training dataset, or manipulate features via projection. These approaches are akin to black-box defences since they make little to no use of the learner.

Filtering instances. These counter-measures assume that the poisoning data in the training dataset or the adversarial samples against the test dataset either exhibit different statistical features, or follow a different distribution. Therefore, they propose to identify and filter out the injected/perturbed data. For example, Laishram & Phoha [59] design Curie that protects SVM classifiers against poisoning attacks that flip labels in the training dataset. Steinhardt, Koh & Liang [95] study the worst-case loss of both data dependent and independent sanitisation methods. Metzen et al. [71] propose to train a detector network that takes input from intermediate layers of a classification network, and filters out adversarial samples. Feinman et al. [39] use (i) kernel density estimates in the feature space of a final hidden layer, and (ii) Bayesian neural network uncertainty estimates to detect the adversarial samples against DNNs. Li et al. [63] apply principal component analysis (PCA) on the convolutional layers of an original convolutional neural network (CNN), and use the extracted statistics to train a cascade classifier that can distinguish between valid and adversarial data.

Injecting data. Goodfellow et al. [43] attribute the existence of adversarial samples to the 'blind spots' of the training algorithm, and propose injecting adversarial examples into training, in order to improve the generalisation capabilities of DNNs [100, 43]—akin to active machine learning in non-adversarial settings. Tramer et al. [101] extend such adversarial training methods by incorporating perturbations generated against other models.

Projecting data. Previous work has shown that high dimensionality facilitates the generation of adversarial samples—owing to an increased attack surface. For example, Wang, Gao & Qi [107] theorise that only one unnecessary feature can 'ruin' the robustness of a model. To counter this,

data can be projected into a lower-dimensional space before testing. Specifically, Bhagoji et al. [22] and Zhang et al. [116] propose defence methods based on dimensionality reduction via PCA, and reduced feature sets, respectively. Das et al. [34] demonstrate that JPEG compression can be used as a pre-processing step to defend against evasion attacks in computer vision, possibly because JPEG compression removes high-frequency signal components, which helps remove imperceptible perturbations. However, these results contradict with Li & Vorobeychik [61] which suggests that more features should be used when facing adversarial evasion.

4.3.2. Learner Robustification

Rather than focusing solely on training and test data, this class of methods—which are white-box in nature—aim to design models to be less susceptible to adversarial samples in the first place.

Stabilisation. Zheng et al. [117] design stability training that modifies the model's objective function by adding a stability term. Their experimental results demonstrate that such modification stabilises DNNs against small perturbations in the inputs. Papernot et al. [87] demonstrate such ideas using a *distillation strategy* against a saliency-map attack. However this has already been proven to be ineffective by Carlini & Wagner [30]. Hosseini et al. [47] propose to improve adversarial training by adding an additional 'NULL' class, and attempt to classify all adversarial samples as invalid.

Moving target. Sengupta et al. [92] apply moving target defences against exploratory attacks: instead of using a single model for classification, the defender prepares a pool of models, and for each image to be classified, one trained DNN is picked following some specific strategy. The authors formulate the interaction as a Repeated Bayesian Stackelberg Game, and show that their approach can decrease the attack's success rate, while maintaining high accuracy for legitimate users.

Robust statistics. Another avenue that has remained relatively unexplored, is to leverage ideas from robust statistics such as *influence functions*, *break-down points*, and *M-estimators* with robust loss functions (such as the Huber loss) that place diminishing cost to increasingly erroneous predictions. Rubinstein et al. [88] were the first to leverage robust statistics in adversarial learning settings for cyber-security, by applying a robust form of PCA that optimises median absolute deviations instead of variance to defend against causative attacks on network-wide volume anomaly detection. Very recently, interest in the theoretical computer science community has turned to robust estimation in high dimensions, e.g. Diakonikolas et al. [35].

4.3.3. Lessons Learned

Despite many proposed defences against machine learning attacks, several recent studies [46, 32] point out that most of these methods (i) unrealistically assume that the attacker is not aware of the defence mechanism, and (ii) only consider relatively weak attacks, e.g. FGSM [43]. Negative results are reported on the effectiveness of these methods against adaptive attackers that are aware of the defence and act accordingly, and against the C&W attack [31] (empirically the most efficient exploratory attack proposed so far).

Specifically, He et al. [46] demonstrate that the following recently proposed methods cannot defend against adaptive adversaries: (i) *feature squeezing* [113], (ii) *specialists+1 ensemble* method [10], (iii)

ensemble of methods in the papers [71, 39, 42]. Carlini & Wagner [32] show that ten detection methods for adversarial samples can be defeated, either by the C&W attack [31], or using white-box/black-box attacks. The authors conclude that the most effective defence so far for DNNs is to apply randomness to the DNN, because it makes generating adversarial samples against the target classifier as difficult as generating transferable adversarial samples.

Much of the most recent work on attacks/defences: (i) are within limited sets of domains—primarily computer vision—and not in security-related domains such as network security, system security, malware analysis, etc. and (ii) regularly omit threat models identifying attacker information, goals and capabilities. It is thus clear that further research is needed to even *understand* the simple question: *Is there an attacker/defender asymmetry in adversarial machine learning, as exists in many other areas of cyber-security*?

5. Project Pillars and Work Packages

The NGTF *Adversarial Machine Learning for Cyber Project* is built on three complementary pillars, each aligned towards one of the three grand challenge aims of Section 1. This section describes the specific work packages comprising these research pillars; Section 6 outlines the proposed project time frames.

5.1. Pillar 1: Autonomous Cyber-Security Operations Platform Demonstrator

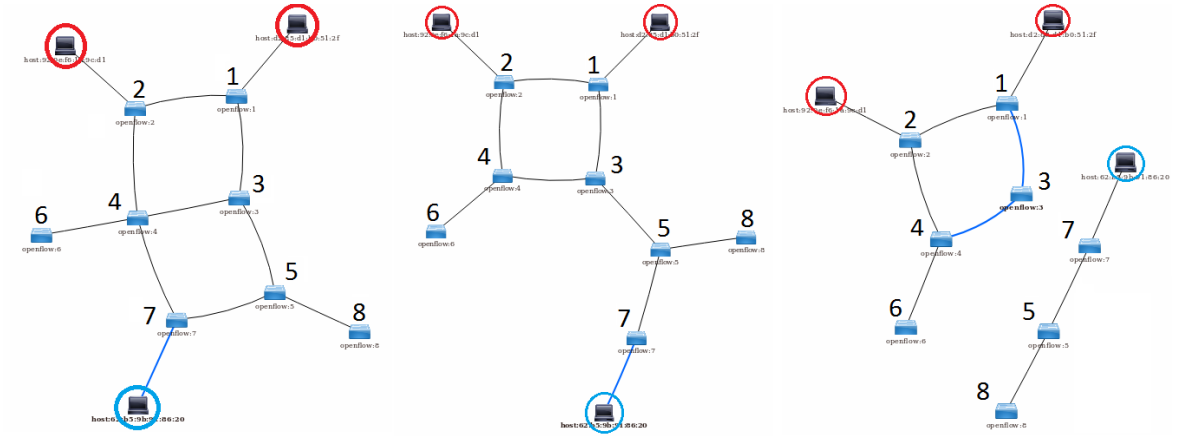
In this pillar, we develop a platform demonstrator and examine how reinforcement learning can be used to enhance the survivability of networks (and, in subsequent pillars, show how to attack and counter-attack the machine learner). Specifically, we focus on the problem of preventing malware from propagating through the network, and penetrating critical services. This section first presents a simplified version of the problem, and demonstrates the feasibility of the RL-based solution. Then we propose the plan for formalising a more realistic setting, and the steps towards solving the problem.

5.1.1. Scoping Result: Proof of Concept Demonstration towards Autonomous Cyber-security Operations Testbed

To determine the feasibility of building a practical testbed for autonomous cyber-security operations (ACO), we developed a proof of concept during the scoping study as described in this section.

Consider the following setup as shown in Figure 5i): A network with three hosts and eight switches is created using Mininet [7], one of the most popular network emulators. OpenDaylight serves as the controller, and monitors the whole-of-network status. Initially, the top two hosts belong to the attacker—they have already compromised the two linked switches, i.e. Switches 1 and 2. The bottom host is the critical server, which the defender should protect from compromise.

We model the task of the defender as an RL problem. In each step, the attacker can further compromise all the nodes that are connected to the nodes under their control, while the defender has perfect



(i) Original topology. *(ii) Removing the first link.* *(iii) Final topology.*

Figure 5: Proof of concept demonstration towards autonomous security operations testbed. OpenDaylight serves as the controller, and monitors the network created using Mininet, where the top two hosts belong to the attacker (circled in red colour), and the bottom one is the critical server (circled in blue colour).

knowledge of the network (i.e. they know whether each node is compromised, and whether each link is turned on/off), but can only turn off one link at a time.

The attacker aims to infect as many nodes as possible, and the defender's goals include (i) protecting as many nodes as possible, (ii) turning off as few links as possible, and (iii) isolating the compromised nodes. More importantly, (iv) the critical node can't be compromised, and (v) the links connecting it with the rest of the network, e.g. the link between the critical server and Node 7 in Figure 5i), cannot be severed as it must complete its critical function with connectivity (i.e.'fight-through in a contested environment'). Each of these goals is translated into a reward.

We implement a basic Q-learning algorithm [99] to decide by which order the links should be turned off. The reader's intuition might be to start with either the link between Switches 1 and 3, or the link between Switches 2 and 4, since Switches 1 and 2 are already infected. However, the Q-learning algorithm chooses to turn off the link between Switches 4 and 7 first (see Figure 5ii)), and then the link between Switches 3 and 5 (see Figure 5iii)). This is because if either of these two links is not turned off, the critical server will be compromised, while turning off any other link is redundant.

5.1.2. Work Package: RL-Powered SDN Platform for Autonomous Cyber-Security Operations (WP 1.1)

The above example is a simple demonstration that RL can be used in SDN to enhance security. Several unrealistic assumptions have been made in the scoping platform, e.g. (i) the defender possesses complete knowledge of the network and attacker status, (ii) their actions are limited to turning off one link per step. We propose to relax these assumptions and explore the problem of autonomous cyber-security operations in more realistic settings.

Objectives: Investigate realistic settings where the defender prevents an attacker from compromising

critical servers, formalised as an RL problem, and develop practical, demonstrable cyber-security solutions.

WP 1.1.1. Formalise the RL problem: Modern SDNs enable intelligent sensors/detectors as dynamically deployed network security measures in order to enforce certain network-wide security policies. Hence, the deployment of detection capabilities should be analysed as an integral part of the network itself. Such detection is closely related to passive network monitoring, where packets are sampled with the intention of finding out more about flows in the network as well as identifying malicious packets, for example. While monitors are only about observation, SDN-based detectors react to security risks directly by dropping (some of the) potentially malicious packets, by sandboxing a suspected insider threat, by re-routing of flows, by raising alarms to human operators, etc. Detectors cannot be fully deployed or activated on all links all of the time due to restrictions on link capacity and link delay. The identification of the strategic points for deploying active network monitors can be computationally complex due to the presence of multiple trade-offs involved. The detector placement problem has also more general implications than security-related ones due to its importance within the context of network management.

We will therefore consider for the testbed a more likely situation where the defender analyses traffic flows at certain nodes, and receives warnings upon detection of abnormal patterns. In this way the problem will be one of partial observability, with false positive and false negative alarms. Additional actions may include on an infected node, isolation, patching, with links re-routed. Once patching is complete, the node could be reconnected, or kept for redundancy. Furthermore, the critical service can be migrated to other servers, at greater cost due to service disruption. Correspondingly, the attacker may focus on a target server, aiming to identify optimal paths to reach it without infecting too many nodes for fear of detection.

In such a situation, the defender's actions may potentially include four categories: (i) configuring anomaly detection at a node, e.g. instrumenting detection (or turning it off), setting alert levels to low/medium/high; (ii) isolating a node and re-routing its links; (iii) reconnecting a node and its links; (iv) migrating the critical server and selecting the destination. Meanwhile, the attacker will carefully choose the nodes to compromise. For example, in the setting of Figure 6, they may infect a node only if it a) is closer to the "backbone" network (switches on the dashed ellipse); b) is in the backbone network; or c) is in the target subset. Table 2 summarises this problem setting.

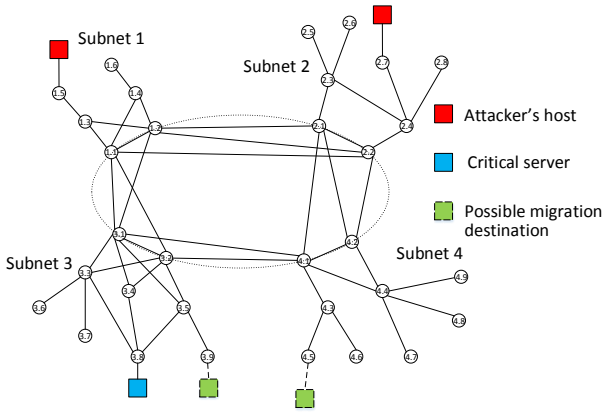


Figure 6: A more complex SDN example with a 'backbone' network and various subnets.

It should be pointed out that the above scenario is only one example, and we will consider other scenarios for further improvements. In addition, while the above description is a RL value-based

approach (see Section 4.1), we will explore other types of solutions as well.

WP 1.1.2. Solution investigation and experimental verification: Once the realistic problem is appropriately formulated, we will compare different kinds of algorithms as red/blue teaming activities, and identify the most efficient algorithm for the given scenario. Classic RL techniques generally rely on hand-crafted representations of sensory input, thus limiting their performance in complex and high-dimensional real world environments. To overcome this limitation, recent developments combine RL techniques with the significant feature extraction and processing capabilities of deep learning models, e.g. deep Q-network (DQN) [72], double DQN [105], and deep deterministic policy gradient algorithm [64]. We will examine these methods first.

In terms of evaluation, we plan to start with synthetic network traffic and Mininet [7], which is a widely used network production emulator that supports experiments with SDN systems. In addition, we will begin with a centralised version of the SDN design, where the controller monitors the entire network.

WP 1.1.3. Evaluation in real security operations settings: After initial evaluation as above, we will extend to hierarchical/distributed designs, where multiple controllers cooperate to prevent attack propagation.

In this later stage of the project, we will switch to real traffic logs. For example, Los Alamos National Laboratory provides a dataset [54] that includes network flow data, along with authentication events, Domain Name Service (DNS) lookups and red teaming events collected over 58 consecutive days. We plan to run this, or a similar, equivalent dataset on real hardware for large scale experiments mimicking as closely as possible, real ACO scenarios.

5.2. Pillar 2: Attacks on Machine Learners

Statistical machine learning techniques have become an increasingly effective tool for cyber defences. While ML deployments may be governed by slow deliberative processes involving testing, security patch deployment, and human-in-the-loop monitoring, learning systems tend to fail when faced with an adaptive adversary attempting to evade detection in a network, or manipulate learned models. Adversaries can systematically probe target networks, pre-plan their attack strategies, and persist over a long period of time inside compromised networks and hosts until they achieve their goal. Understanding potential vulnerabilities introduced by deploying ML systems requires active research into attacks on machine learners, the focus of this second pillar.

5.2.1. Objectives

In this project we will consider the following three settings for reinforcement learning attacks, namely, *white-box*, *black-box*, and *distributed* settings, the latter being a situation where, for example, several adversaries collude to conduct an attack. We will characterise how the effectiveness of adversarial examples is impacted by the knowledge of an adversary of the RL policy network (white-box vs. black-box), as well as the number of adversaries (single agent or distributed), when deep RL algorithms are used to learn the policy. More specifically, we will investigate the following problems for each of these settings:

1. **Effectiveness of Existing Adversarial Examples —** Several strategies for designing adversarial

examples[4] have been introduced for supervised learning (and image processing in particular). We will investigate how we can apply these approaches to deep RL and how effective they are in other contexts, such as SDN and Internet of Things (IoT) devices. Specifically, in order to quantify the effectiveness of these strategies, we will define appropriate metrics that consider factors including the required changes, e.g. in terms of the distance between the perturbed and original data, the frequency of launching the attack, the success rate, etc.

2. **Impact of Knowledge of Adversary —** We will characterise how the effectiveness of adversarial examples is impacted by the knowledge of an adversary of the policy network (white-box vs. black-box).
3. **Impact of Number of Adversaries —** We will characterise how the effectiveness of adversarial examples is impacted by the number of adversaries (centralised vs. distributed attacks).
4. **Bounds on Adversarial Influence —** We will aim to model upper and lower bounds of an adversary's influence on a learner under various threat models.
5. **Developing New Attacks —** We will develop new attacks, such as boiling-frog attacks [88], that can significantly affect the behaviour of deep RL without being apparent to the target. To minimise the number of adversarial samples and adversaries that are apparent, we will incorporate a guideline system that can assist the attacker to choose the most appropriate time for injecting an adversarial example – hence minimising the number of adversarial samples, while ensuring the effectiveness of the attack. The attack model will also be armed with a prediction model and reasoning model that can enable the adversary to operate in highly dynamic environments (where the target may change its defence strategy very quickly). We will also investigate how the attack can be tailored to different levels of knowledge and cooperation for the adversary.

5.2.2. Work Package: Metrics in Adversarial Machine Learning (WP 2.1)

In order to evaluate effectiveness of attacks on machine learning, and subsequent defences, it is critical to have appropriate metrics.

Objectives: In this work package, we will investigate metrics for evaluation of attacks and defences in adversarial machine learning. A key objective for these metrics is to reflect real-world threat models in Defence domains.

WP 2.1.1. Cost of Attacks: To characterise the feasibility and effectiveness of attacks on learning systems, we will explore several domains relevant to Defence such as cyber-security, malware detection, computer vision, taking note of real-world threat models. For example, what adversarial examples are feasible? On what basis do adversarial perturbations cost the attacker?

5.2.3. Work Package: White-Box RL Attacks (WP 2.2)

In this second work package of the learning attacks pillar, we will investigate the white-box setting, which studies an omniscient and omnipotent adversary with access to: (i) the training data, (ii) the

[4] Adversarial examples are crafted input samples that are small but intentionally worst-case perturbations to examples from the dataset, such that the perturbed input results in the machine learning model outputting an incorrect answer with high confidence [43].

feature set, (iii) the target policy, (iv) the learning algorithm and the kind of decision function, (v) the classifier's decision function and its parameters, and (vi) the feedback available from the learner.

Objectives: We will carry out the following tasks in these white-box settings.

WP 2.2.1. Evaluating Attack Effectiveness: We will investigate the vulnerability of reinforcement learning to existing supervised learning attacks, including the Fast Gradient Sign Method (FGSM) [43], Basic Iterative Method (BIM) [58], Jacobian-based Saliency Map Attack (JSMA) [86], and Carlini & Wagner (C&W) [31]. While these attacks have been demonstrated to be able to disrupt supervised learning and RL based computer vision applications, their effectiveness in other contexts like network security, IoT or malware detection is still an open question.

In IoT systems, for example, adversaries can constantly corrupt the sensor readings, or interfere with communication between devices, in order to prevent the agent from learning the optimal policy, or take significantly more time to converge. One point worth mentioning is that even though experience replay [72]—keeping a buffer of past experiences, and randomly sampling a batch for training—brings a number of advantages, it may also be exploited by adversaries—the corrupted experiences may affect multiple training episodes.

Another potential area is evading machine learning malware detection [14], where the agent receives a feature vector of a malware sample, i.e. a state s from the environment, and takes a certain action that does not break the PE file format or the malware's intended functionality. If the modified sample passes the detection system (a given anti-malware engine), the reward is 1 (i.e. the malware is deemed to be malicious). Otherwise, if the malware is deemed benign, the reward is 0. Similar to the previous research in the vision domain, we intend to find the optimal balance between minimising the perturbations and maximising the attack's success rate.

WP 2.2.2. Boiling-Frog Attack: We will devise approaches to boiling-frog attacks for deep RL (DRL) networks, where an adversary gradually increases the level of injected noise in the samples, and hence decreases the chance that the poisoning activity itself will be detected. We will *craft adversarial samples* by using the knowledge extracted from a guiding mechanism to increase the amplitude of the target's model gradients. If adversarial gradients are high, crafting adversarial samples becomes easier because small perturbations will induce large output variations. We propose to explore methods for guiding adversarial sampling, such as:

1. Perturbing every sample,
2. Only injecting an adversarial sample every N intervals (e.g. minutes), whilst not perturbing the intermediate frames, and
3. Using the RL policy's value function (computed over the original input) as a guide to find the most effective time for injecting perturbed samples.

We will investigate how effective such guiding mechanisms are with respect to the different magnitudes and different types of perturbations discussed in WP 2.2.1. The success of such boiling-frog attacks can then be evaluated using notions of a predictive model and a reasoning model.

In a highly dynamic environment with adaptive sources, an adversary requires the ability to predict the

target's behaviour. Monitoring slight variations in the target's behaviour/strategy over a sequence of interactions could be facilitated with the following models constructed by the attacker:

1. Predictive model: to predict the target's next action.
2. Reasoning model: to choose a suitable best response by analysing its own average reward (whether it is losing or winning).

5.2.4. Work Package: Black-Box RL Attacks (WP 2.3)

In black-box attacks, an adversary does not have access to the initialisation of the target policy, nor knows what learning algorithm is used. The attacker only has access to the outputs. In this setting, the adversary will typically train a surrogate (white-box) model on either the black-box outputs, or on related but not identical training data to that which was used to train the target. With the surrogate model in hand, the attacker is then free to generate attack examples. Therefore, the black-box setting represents a more realistic threat model. Moreover with partial information about the makeup of the target model, an attacker may exercise a grey-box attack where any knowledge possessed may go towards constraining the surrogate model, surrogate training set, feature set selection, etc. When the target's reward function is not known, it can be estimated via Inverse Reinforcement Learning techniques [118][5]. The only thing that the adversary can directly manipulate is the configuration of the environment observed by the target.

The main challenge in this context is to generate a robust adversarial RL model, since the gap between simulation and the real world can be so large that policy-learning approaches fail to transfer. Also, even if policy learning is done in the real world, the scarcity of data can lead to failed generalisations from the adversary to the target.

Objectives: In this work package, we will extend WPs 2.2.1–2.2.2 to the black-box setting (i.e. WPs 2.3.1–2.3.2), and address the transferability issue (and improve the generalisation of an adversary's attack) by also carrying out the following task (WP 2.3.3).

WP 2.3.3. Attack Transferability: To assess transferability of adversarial samples, we will explore the dimensionality of the space of adversarial inputs. Adversarial examples generally occur in large, contiguous regions, rather than being scattered randomly in small pockets. The dimensionality of these subspaces is directly relevant to the transferability problem: the higher the dimensionality, the more likely it is that the subspaces of two models will intersect significantly [102]. We will also explore other factors in transferability such as surrogate and target generalisation error, smoothness properties, etc.

5.2.5. Work Package: Distributed (Collaborative) RL Attacks (WP 2.4)

Here we aim to investigate the impact of distributed (collaborative) attacks, whereby several adversaries cooperate (using a joint optimisation function) to launch an attack. In this setting, each adversary is able to inject perturbed samples at a lower frequency, and correspondingly the target will have greater difficulty in identifying the attackers. The challenge is to ensure that the knowledge is continuously

[5]Inverse Reinforcement Learning (IRL) considers the problem of extracting a reward function from observed behavior of an expert acting in an environment.

shared, refined, and concluded from all adversaries' perspectives.

Objectives: In this work package we will develop a distributed deep RL architecture, and extend WPs 2.2.1–2.2.2 and 2.3.1–2.3.3 to this architecture.

WP 2.4.1. Distributed Architecture: We will first design a distributed multi-agent deep RL algorithm that learns (1) how to handle the interactions between multiple adversaries, and (2) how to jointly optimise the policies of all adversaries. When more than one attack is conducted (on one or more targets) by the adversaries, this may involve multi-task learning with distributed deep reinforcement learning. In such a case, the distributed algorithm may support: (i) role specialisation, (ii) hierarchical communication, (iii) task decomposition, and (iv) reward shaping.

WP 2.4.2. Distributed Adversarial Guidance & Attack Evaluation: We will develop a guidance mechanism that jointly selects appropriate *time and adversary* for each adversarial example. In this architecture, the concept of the individual optimisation loses its meaning, because the repayment of each adversary not only depends on itself but also on the choice of other adversaries. Our approach may again involve tailoring fast learning to the distributed attack architecture, so that it is able to generate a predictive model and reason about multiple adversaries. The challenge in this context is in integrating information across agents and the environment. Finally, the various approaches for the distributed setting will be evaluated.

5.3. Pillar 3: Adversarial Machine Learning Defences

There exists a fundamental relationship between security, decision making, and adversarial machine learning. Theoretical models at the system level play an increasingly important role in these fields as they mature from their earlier qualitative and empirical natures. An alternative to implicit and heuristic decision-making is the analytical approach based on mathematical models. A good example is a packet filtering system deciding on whether to drop a packet or not based on a preset threshold. Instead of setting such a threshold heuristically, (dynamically) determining its value as a decision criterion can be investigated analytically and solved within an optimal and robust control framework based on given preferences. The quantitative approach described has multiple advantages over the ad-hoc alternative, e.g. the knowledge of the decision-maker can be expressed through mathematical models in a transparent and durable manner or the decision-making framework can be applied to multiple similar problems.

5.3.1. Work Package: Formulating Security Games for Adversarial Machine Learning (WP 3.1)

Security games provide an analytical framework for modelling the interaction between malicious attackers, who aim to compromise cyber-based systems, and owners or administrators defending them. In the AMLC setting, the strategic struggle over the control of the ML scheme in the form of data distortion and robust decisions, and the associated interaction between attackers and defenders is formalised using the rich mathematical basis provided by the field of game theory.

Objectives: Formulate security games for AMLC capturing various aspects of attacks and defences

under relevant threat models.

WP 3.1.1. Identify the components for the security game, e.g. objectives & constraints for the players, appropriate solution concepts for AML: The underlying idea behind the game-theoretic models in security is the allocation of limited available resources from both players' perspectives. If the attacker and defenders had access to unlimited resources (e.g. time, computing power, data), then the solutions of these games would be trivial and the contribution of such a formalisation would be limited. In reality, however, both attackers and defenders have to act strategically and make numerous decisions when allocating their respective resources—as governed by an identified threat model. For example, within the context of AMLC, the attackers have to be subtle and covert in order not to be immediately detected and defenders have to be mindful of overall system performance when countering various attacks.

A security game can be defined with four components: *the players*, the set of *possible actions* for each player, the *outcome* of each player interaction (action-reaction), and *information structures* in the game. In a *two-player* security game in the AMLC setting, one player is the *attacker* which abstracts one or multiple entities with malicious intent to compromise the defended system, in this case the ML algorithm. Defining a single metaphorical "attacker" player simplifies the analysis at the first iteration. The other player represents the ML algorithm which needs to perform satisfactorily in the presence of adversaries.

5.3.2. Work Package: Stochastic Security Games and Reinforcement Learning (WP 3.2)

Stochastic or Markov security games use probability theory to model the unknown and uncontrollable parameters in security problems such as AML. Although they are more complex due to their mathematical sophistication, Markov games enable study of the interaction between attackers and defenders in a more realistic way. Moreover, there is a direct relationship from these ideas to reinforcement learning, e.g. Q-learning [12].

Objectives: Develop stochastic security games and associated RL formulations, especially making use of novel deep learning methods.

Stochastic games are played between the attacker and the defender on a *state space* representing the environment. A *state* may be an operational mode of the networked system such as which units are operational, active countermeasures, or whether parts of the system are compromised. In the adopted stochastic model, the states evolve probabilistically according to a defined stochastic process with the Markov property. The Markov property holds naturally in many systems and provides a useful simplification for others. The resulting stochastic process is parameterized by player actions allowing modeling of the effect of player decisions on the networked system properties. For example, the probability of detection of a specific attack is a function of attacker behavior, e.g. intensity of attack or whether the system is targeted, as well as the amount of monitoring resources allocated by the defender.

In addition to providing the defence system guidelines for countermeasures and resource allocation, stochastic security games aim to analyse the *behaviour of rational attackers*. Within the framework of stochastic security games, attacker behavior is represented as a probability distribution over possible attacks (actions) in each state. Attacker strategies can be derived under various assumptions and for

different scenarios resulting in Nash equilibrium and other solutions.

The detector placement problem discussed in Pillar 1 (see WP 1.1.1) can be investigated adopting a game-theoretic approach, more specifically the stochastic game framework discussed here. In the worst-case scenario, the attackers attempting to gain unauthorized access to a target system residing in the network or compromise its accessibility through distributed denial of service (DDoS) attacks are expected to have complete knowledge of the internal configuration of the network such as routing states or detector locations. Thus, the attackers should be seen as rational and intelligent players who respond to defensive actions by choosing different targets or routes to inject the malware. Due to this presumed adaptive behavior of the opponent, dynamic defensive measures should be considered that take into account the actions of the attackers. Otherwise, the detector placement and activation algorithms may yield suboptimal results as a consequence of the attackers circumventing them through intelligent routing.

WP 3.2.1. Investigate limited information and full information settings: In an AMLC setting, the players are direct adversaries, which can be modeled as zero-sum games. Multi-agent Markov decision processes, such as dynamic programming methods derived from Markov Decision Processes (MDPs), are utilised for solving these types of games. When limitations are imposed on information available to players in stochastic security games, they can adopt various learning schemes. Such games are said to be of *limited information* due to the fact that each player observes the other players' moves and the evolution of the underlying system only partially or indirectly. The players refine in such cases their own strategies online by continuously learning more about the system and their adversaries. Consequently, they base decisions on limited observations by using for example *Q-learning* methods.

In more realistic settings, the games considered are not ones of full information due to the fact that each player observes the other players' moves and the evolution of the underlying system sometimes only partially and indirectly. Therefore, the players have to base their decisions on their own occurring costs and limited observations of the system and other players' actions. Specifically, the players optimise their strategies with decreasing information available to them using methods such as value iteration to solve MDPs, minimax-Q, and naive Q-learning, the latter method being a direct form of reinforcement learning.

In the full information case each player knows everything about the AMLC setup as well as the preferences and past actions of its opponent. Hence, players may utilise well-known MDP methods such as value iteration to calculate their own optimal mixed strategy solutions to the zero-sum game. When we relax the assumption of perfect knowledge, the attacker has to choose its strategy without knowing the characteristics of the environment represented by the transition probabilities. In this case, the attacker can calculate its optimal strategy online (i.e. while playing the game) using a technique called minimax-Q, which is a variation of the standard Q-learning technique. In both of these cases, the players know each other's costs and observe their opponent's past actions. In a more realistic AMLC setting, a player only observes the environment and keeps track of its own actions and costs. In this third case, single agent "naive" Q-learning (ignoring the other player's actions) has been used as a possible approach in the literature [12].

WP 3.2.2. Multi-agent Reinforcement Learning (MARL): Up until now we have only considered single-agent reinforcement learning. However, in complex systems such as distributed SDN, traffic

control, autonomous vehicles and power grids, due to either scale or distributed nature, it is beyond any single agent's capacity to manage the whole system. Instead, a set of autonomous agents interact in the shared environment and potentially with each other, in order to achieve the global goal [27, 25]. For example, different types of agents in power grids, including generator agent, switch agent, load agent and scheduler agent, exchange local information to minimise the restoration time after catastrophic disturbances [114].

In MARL, all agents are learning simultaneously, and their actions depend on each other. Therefore, each agent is facing a moving-target learning problem. In addition, their rewards are correlated, and cannot be maximised independently. All of these make MARL problems more challenging, in terms of: (1) the curse of dimensionality, (2) setting appropriate goals, (3) non-stationarity and (4) the need for coordination [27].

We intend to model a MARL problem as a stochastic game, and define it as a tuple $\langle S, A_1, ..., A_n, p, R_1, ..., R_n \rangle$, where n is the number of agents; S is the set of environment states; $A_i, i = 1, ..., n$ is the i^{th} agent's own action set, and altogether they form the joint action set $A = A_1 \times ... \times A_n$; with $p : S \times A \times S \rightarrow [0, 1]$ being the state transition probability function; and $R_i, i = 1, ..., n$ the reward function for the i^{th} agent.

Specifically, we will focus on the mixed MARL scenario, where the reward function for each agent can be different, but the agents are not competing. A number of algorithms have been proposed for this type of problem, including Non-Stationary Converging Policies (NSCP) [110], Win-or-Learn-Fast Policy Hill-Climbing (WoLF-PHC) [26] and Extended Optimal Response (EXORL) [96]. We aim to investigate these methods and improve upon them. It is worth noticing that the coordination between agents may create a new source of attack surface [19]. For example, the adversaries can either passively eavesdrop the shared information, or actively spoof/compromise agents, and then falsify signals or refuse to share information at certain times. Therefore, security will be one of our main concerns when we design new solutions for a MARL problem.

5.3.3. Work Package: Robust Statistics (WP 3.3)

In addition to game theory, adversarial learning has strong connections with the field of robust statistics [51]. A classic problem setting studied in the latter is the Tukey-Huber contamination model [104, 51], wherein there is some distribution P of interest, but one observes samples from a mixture distribution $\tilde{P} = (1 - \epsilon) \cdot P + \epsilon \cdot Q$, where Q is an arbitrary distribution that may have no relation to P. One can think of Q as comprising outliers that are potentially adversarially chosen, with the observed sampling distribution $\tilde{P}$ being corrupted or compromised owing to the injection of outliers from Q.

Considerable research has focussed on reliable estimation of P under this model, which provides a building block for applications in cyber-security such as volume-wide anomaly detection [88]. Estimation of P may be made under the assumption that it belongs to a simple parametric family, such as a univariate Gaussian [50]. A key idea is the use of suitable medians rather than means, owing to the robustness properties of the former. While this idea can be extended to higher dimensions via the Tukey depth [103], such approaches face a strong computational barrier: the Tukey depth is NP-hard to compute [53]. This computational barrier unfortunately affects a broader class of robust estimators [21].

While the above appears to paint a pessimistic picture, reassuringly, progress is possible. In the computer science literature, Diakonikolas et al. [35] recently provided efficient algorithms for robust parameter estimation. Specifically, they consider the following problem: given samples from a corrupted version of a distribution P, the goal is to produce an estimate $\hat{P}$ such that $\|P - \hat{P}\|_{\mathrm{TV}}$ is small, where $\|\cdot\|_{\mathrm{TV}}$ denotes the total-variation distance between probability distributions. Their procedure relies on three steps:

1. **Find an appropriate parameter distance**. We must pick a means of measuring distance between the parameters of our estimate and the true distribution. For example, in the case of estimating the mean of a (multi-dimensional) Gaussian, it is convenient to simply pick the Euclidean distance between the true and predicted means μ, $\hat{\mu}$.
2. **Detect when the naïve estimator is compromised**. A key step is to determine if there is in fact corruption in the observed sample, and if so, which data points are affected. This relies on a crucial insight: an adversary cannot modify the mean of a distribution without some of the corrupted points being far from the mean in some direction.
3. **Find good parameters, or make progress**. Having determined the affected data points, one can either apply a filtering procedure to remove them, or suitably weight them to ensure an accurate estimate.

There are several interesting directions to build upon this line of fundamental and relevant work.

WP 3.3.1. Natural parameter estimation: Suppose we are interested in estimating a distribution coming from the exponential family (which includes the Gaussian distribution). Rather than focus on estimation of this distribution's standard parameters, one can instead look at estimation of its *natural parameters*. These parameters play a key role in information-geometric study of probability distributions, and have seen application in for example the natural gradient-descent procedure [13].

WP 3.3.2. Robust learning algorithms: The above procedure provides an algorithm to solve a precise problem: robust estimation of a distribution. However, what can be said about making an *algorithm* itself robust? As a concrete candidate, consider the problem of stochastic gradient descent for online or large-scale minimisation of a loss function (as is typically used in machine learning). Can one determine corruption of samples during operation of this procedure, and correct for this? A natural thought is that, analogous to the use of the second moment to detect the corruption of points in the parameter-estimation problem, one might look at the values of the loss function being optimised. Formalising this idea is an interesting challenge, with solutions likely very impactful.

5.3.4. Work Package: Min-Max Learning Formulations (WP 3.4)

Consider a supervised learning problem where one has labelled samples $\{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$, and the labels $\mathcal{Y}$ are believed to be corrupted by an adversary. One would like to design a learning procedure that can make accurate predictions about the labels on future instances.

While many approaches to this problem are possible, an elegant and computationally tractable one is the following, explored in various guises [44, 16, 108, 38]. Let us assume that the adversary generates labels according to some distribution $\mathbb{P}_{\mathrm{adv}}(\mathcal{Y} \mid \mathcal{X})$, which is constrained in the following way: for a fixed feature mapping $\Phi\colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^D$, the mean under the adversarial distribution must match the

mean under the empirical distribution, i.e.

$$\mathop{\mathbb{E}}_{(\mathsf{X},\mathsf{Y}_{\text{adv}})}[\Phi(\mathsf{X},\mathsf{Y}_{\text{adv}})] = \frac{1}{N}\sum_{i=1}^{N}\Phi(x_i, y_i).$$

One may then seek a classification model $f : \mathcal{X} \to \mathcal{Y}$ that ensures minimal loss under the *worst-possible* feasible $\mathbb{P}_{\text{adv}}(\mathcal{Y} \mid \mathcal{X})$, i.e.

$$\min_{f} \max_{\mathbb{P}_{\text{adv}}} \mathop{\mathbb{E}}_{(\mathsf{X},\mathsf{Y}_{\text{adv}})}[\mathcal{L}(\mathsf{Y}_{\text{adv}}, f(\mathsf{X}))]$$

for a given loss function $\mathcal{L}$. This can be seen as a *min-max* learning problem, leading to an interpretation of the adversarial learning problem as a two-player game.

For the logarithmic loss, this procedure leads to the family of log-linear models [106]. Interestingly, when applied to certain non-convex loss functions, the procedure results in a convex objective; thus, guarding against adversarial perturbation is equivalent to smoothing the objective used for learning. For example, Asif et al. [16] showed that for cost-sensitive losses, the objective can be cast as the solution to a linear program. For the 0-1 loss, Farnia & Tse [38] showed that the problem is equivalent to minimising a modified version of the hinge loss underpinning a Support Vector Machine (SVM).

There are several interesting directions to build upon this work.

WP 3.4.1. Application to label noise: The above framework restricts the power of the adversary by constraining the expectation of certain features. One can conceive of other forms of plausible constraints on the adversary. For example, consider a binary classification setting where the adversary is allowed to flip labels with an instance-dependent probability; we might then constrain that on average, this flipping function cannot be too large. Such constraints naturally arise in threat models in which attackers value stealth. Can one adapt the above to yield a simple procedure to cope with label noise? It is possible that for example this may yield smoothed versions of losses that arise when one knows *a priori* the flipping function [78].

WP 3.4.2. Generalisation to complex performance measures: At its core, the above framework is concerned with losses that decompose as averages over individual instances. Many real-world performance measures do not however have this property; in the binary case, the F-score and area under a Receiver Operating Curve (ROC) are two simple but popular examples. Can one adapt the framework to handle such performance measures? For the area under the ROC, progress is likely possible owing to its interpretation as a 0-1 loss over pairs [11]; however, the precise form of the resulting loss is of interest. For measures such as F-score, a fruitful direction might be the variational cost-sensitive representation of these measures [77].

6. Project Timetable

The NGTF AMLC project plan timetable is given in Table 3. The timeline is presented in six-monthly phases (H1 and H2) for each calendar year.

References

[1] OpenFlow. http://archive.openflow.org/wp/learnmore/, 2011.

[2] SDN architecture. Technical report, June 2014.

[3] Diabetic Glucose Monitoring Becoming More Intelligent and Connected. https://newsroom.intel.com/editorials/diabetic-glucose-monitoring-becoming-intelligent-connected/, 2015.

[4] Open vSwitch. http://openvswitch.org/, 2016.

[5] Cisco Open SDN Controller. https://www.cisco.com/c/en/us/products/cloud-systems-management/open-sdn-controller/index.html, 2017.

[6] Floodlight OpenFlow Controller. http://www.projectfloodlight.org/floodlight/, 2017.

[7] Mininet: An Instant Virtual Network on your Laptop. http://mininet.org/, 2017.

[8] NOX/POX. https://github.com/noxrepo, 2017.

[9] OpenDaylight. https://www.opendaylight.org/, 2017.

[10] M. Abbasi and C. Gagné. Robustness to Adversarial Examples through an Ensemble of Specialists. *eprint arXiv:1702.06856*, 2017.

[11] S. Agarwal, T. Graepel, R. Herbrich, S. Har-Peled, and D. Roth. Generalization Bounds for the Area Under the ROC Curve. *J. Mach. Learn. Res.*, 6:393–425, Dec. 2005.

[12] T. Alpcan and T. Basar. An intrusion detection game with limited observations. In *12th International Symposium on Dynamic Games and Applications*, Sophia Antipolis, France, July 2006.

[13] S. I. Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10(2):251–276, Feb. 1998.

[14] H. Anderson, A. Kharkar, B. Filar, and P. Roth. Evading machine learning malware detection. In *Black Hat USA*, 2017.

[15] W. Ashford. Adversarial machine learning tops McAfee's 2018 security forecast. *ComputerWeekly.com*, Nov. 2017.

[16] K. Asif, W. Xing, S. Behpour, and B. D. Ziebart. Adversarial Cost-sensitive Classification. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI'15, pages 92–101, Arlington, Virginia, United States, 2015. AUAI Press.

[17] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar. The Security of Machine Learning.

Machine Learning, 81(2):121–148, Nov. 2010.

[18] L. Beaudoin. *Autonomic computer network defence using risk states and reinforcement learning*. PhD thesis, University of Ottawa (Canada), 2009.

[19] V. Behzadan and A. Munir. Models and Framework for Adversarial Attacks on Complex Adaptive Systems. *arXiv:1709.04137 [cs]*, Sept. 2017.

[20] V. Behzadan and A. Munir. Vulnerability of Deep Reinforcement Learning to Policy Induction Attacks. *eprint arXiv:1701.04143*, Jan. 2017.

[21] T. Bernholt. Robust estimators are hard to compute. Technical report, Universität Dortmund, 2006.

[22] A. N. Bhagoji, D. Cullina, and P. Mittal. Dimensionality Reduction as a Defense against Evasion Attacks on Machine Learning Classifiers. *arXiv:1704.02654*, 2017.

[23] B. Biggio, I. Corona, B. Nelson, B. I. Rubinstein, D. Maiorca, G. Fumera, G. Giacinto, and F. Roli. Security evaluation of support vector machines in adversarial environments. In *Support Vector Machines Applications*, pages 105–153. Springer International Publishing, 2014.

[24] B. Biggio, B. Nelson, and P. Laskov. Poisoning Attacks against Support Vector Machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1467–1474, Edinburgh, Scotland, 2012. Omnipress.

[25] D. Bloembergen, K. Tuyls, D. Hennes, and M. Kaisers. Evolutionary Dynamics of Multi-agent Learning: A Survey. *J. Artif. Int. Res.*, 53(1):659–697, May 2015.

[26] M. Bowling and M. Veloso. Multiagent Learning Using a Variable Learning Rate. *Artificial Intelligence*, 136(2):215–250, Apr. 2002.

[27] L. Buşoniu, R. Babuška, and B. De Schutter. Multi-agent Reinforcement Learning: An Overview. In D. Srinivasan and L. C. Jain, editors, *Innovations in Multi-Agent Systems and Applications - 1*, pages 183–221. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. DOI: 10.1007/978-3-642-14435-6_7.

[28] C. Burkard and B. Lagesse. Analysis of Causative Attacks Against SVMs Learning from Data Streams. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, IWSPA '17, pages 31–36, New York, NY, USA, 2017. ACM.

[29] S. Cao, Y. Wang, and C. Xu. Service Migrations in the Cloud for Mobile Accesses: A Reinforcement Learning Approach. In *2017 International Conference on Networking, Architecture, and Storage (NAS)*, pages 1–10, Aug. 2017.

[30] N. Carlini and D. Wagner. Defensive Distillation is Not Robust to Adversarial Examples. *arXiv:1607.04311*, 2016.

[31] N. Carlini and D. Wagner. Towards Evaluating the Robustness of Neural Networks. *eprint arXiv:1608.04644*, 2016. arXiv: 1608.04644.

[32] N. Carlini and D. Wagner. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods. *eprint arXiv:1705.07263*, 2017. arXiv: 1705.07263.

[33] S. P. Chung and A. K. Mok. Advanced Allergy Attacks: Does a Corpus Really Help? In *Recent Advances in Intrusion Detection*, Lecture Notes in Computer Science, pages 236–255. Springer, Berlin, Heidelberg, Sept. 2007.

[34] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau. Keeping the Bad Guys Out: Protecting and Vaccinating Deep Learning with JPEG Compression. *eprint arXiv:1705.02900*, May 2017. arXiv: 1705.02900.

[35] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust Estimators in High Dimensions without the Computational Intractability. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 655–664, Oct. 2016.

[36] M. Duggan, J. Duggan, E. Howley, and E. Barrett. A reinforcement learning approach for the scheduling of live migration from under utilised hosts. *Memetic Computing*, pages 1–11, Dec. 2016.

[37] I. Evtimov, K. Eykholt, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati, and D. Song. Robust Physical-World Attacks on Machine Learning Models. *arXiv:1707.08945*, 2017.

[38] F. Farnia and D. Tse. A Minimax Approach to Supervised Learning. In *Advances in Neural Information Processing Systems*, pages 4233–4241, 2016.

[39] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner. Detecting Adversarial Samples from Artifacts. *eprint arXiv:1703.00410*, 2017.

[40] J. Finkle. J&J warns diabetic patients: Insulin pump vulnerable to hacking. *Reuters*, Oct. 2016.

[41] S. Fogarty. 7 Essentials Of Software-Defined Networking. URL address: http://www.networkcomputing.com/cloud-infrastructure/7-essentials-software-defined-networking/1672824201, Nov. 2013.

[42] Z. Gong, W. Wang, and W.-S. Ku. Adversarial and Clean Data Are Not Twins. *eprint arXiv:1704.04960*, 2017.

[43] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and Harnessing Adversarial Examples. *eprint arXiv:1412.6572*, 2014.

[44] P. D. Grünwald and A. P. Dawid. Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. *The Annals of Statistics*, 32(4):1367–1433, 2004.

[45] Y. Han and B. I. P. Rubinstein. Adequacy of the Gradient-Descent Method for Classifier Evasion

Attacks. *arXiv:1704.01704*, Apr. 2017. arXiv: 1704.01704.

[46] W. He, J. Wei, X. Chen, N. Carlini, and D. Song. Adversarial Example Defenses: Ensembles of Weak Defenses are not Strong. *eprint arXiv:1706.04701*, 2017.

[47] H. Hosseini, Y. Chen, S. Kannan, B. Zhang, and R. Poovendran. Blocking Transferability of Adversarial Examples in Black-Box Learning Systems. *eprint arXiv:1703.04318*, 2017.

[48] R. Huang, X. Chu, J. Zhang, and Y. H. Hu. Energy-efficient Monitoring in Software Defined Wireless Sensor Networks Using Reinforcement Learning: A Prototype. *Int. J. Distrib. Sen. Netw.*, 2015:1:1–1:1, Jan. 2015.

[49] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. Adversarial Attacks on Neural Network Policies. *eprint arXiv:1702.02284*, 2017.

[50] P. J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

[51] P. J. Huber. *Robust Statistics*. Wiley, 1981.

[52] V. M. Igure, S. A. Laughter, and R. D. Williams. Security issues in SCADA networks. *Computers & Security*, 25(7):498–506, 2006.

[53] D. S. Johnson and F. P. Preparata. The densest hemisphere problem. *Theoretical Computer Science*, 6(1):93 – 107, 1978.

[54] A. Kent. Comprehensive, Multi-Source Cyber-Security Events. https://csr.lanl.gov/data/cyber1/, 2015.

[55] S. Kim, J. Son, A. Talukder, and C. S. Hong. Congestion prevention mechanism based on Q-leaning for efficient routing in SDN. In *2016 International Conference on Information Networking (ICOIN)*, pages 124–128, Jan. 2016.

[56] P. W. Koh and P. Liang. Understanding Black-box Predictions via Influence Functions. *arXiv:1703.04730 [cs, stat]*, Mar. 2017. arXiv: 1703.04730.

[57] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[58] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial Machine Learning at Scale. *arXiv:1611.01236 [cs, stat]*, Nov. 2016. arXiv: 1611.01236.

[59] R. Laishram and V. V. Phoha. Curie: A method for protecting SVM Classifier from Poisoning Attack. *arXiv:1606.01584 [cs]*, June 2016.

[60] J. Leyden. Kaspersky Lab denies tricking AV rivals into nuking harmless files. *The Register*, Aug. 2015.

[61] B. Li and Y. Vorobeychik. Feature Cross-substitution in Adversarial Classification. In *NIPS*, NIPS'14, pages 2087–2095, Cambridge, MA, USA, 2014. MIT Press.

[62] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik. Data Poisoning Attacks on Factorization-Based Collaborative Filtering. *eprint arXiv:1608.08182*, 2016.

[63] X. Li and F. Li. Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics. *arXiv:1612.07767 [cs]*, Dec. 2016.

[64] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv:1509.02971 [cs, stat]*, Sept. 2015.

[65] S. C. Lin, I. F. Akyildiz, P. Wang, and M. Luo. QoS-Aware Adaptive Routing in Multi-layer Hierarchical Software Defined Networks: A Reinforcement Learning Approach. In *2016 IEEE International Conference on Services Computing (SCC)*, pages 25–33, June 2016.

[66] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun. Tactics of Adversarial Attack on Deep Reinforcement Learning Agents. *eprint arXiv:1703.06748*, Mar. 2017.

[67] H. Mao, M. Alizadeh, I. Menache, and S. Kandula. Resource Management with Deep Reinforcement Learning. In *15th ACM Workshop on Hot Topics in Networks*, HotNets '16, pages 50–56, New York, NY, USA, 2016. ACM.

[68] J. Medved, R. Varga, A. Tkacik, and K. Gray. OpenDaylight: Towards a Model-Driven SDN Controller architecture. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6, June 2014.

[69] S. Mei and X. Zhu. Using Machine Teaching to Identify Optimal Training-set Attacks on Machine Learners. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2871–2877, Austin, Texas, 2015. AAAI Press.

[70] A. Mestres, A. Rodriguez-Natal, J. Carner, P. Barlet-Ros, E. Alarcón, M. Solé, V. Muntés-Mulero, D. Meyer, S. Barkai, M. J. Hibbett, G. Estrada, K. Ma'ruf, F. Coras, V. Ermagan, H. Latapie, C. Cassar, J. Evans, F. Maino, J. Walrand, and A. Cabellos. Knowledge-Defined Networking. *SIGCOMM Comput. Commun. Rev.*, 47(3):2–10, Sept. 2017.

[71] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On Detecting Adversarial Perturbations. *eprint arXiv:1702.04267*, 2017.

[72] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, Feb. 2015.

[73] A. W. Moore and C. G. Atkeson. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13(1):103–130, Oct. 1993.

[74] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage. Inferring Internet Denial-of-service Activity. *ACM Trans. Comput. Syst.*, 24(2):115–139, May 2006.

[75] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal Adversarial Perturbations. *eprint arXiv:1610.08401*, 2016.

[76] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. In *CVPR*, pages 2574–2582, 2016.

[77] H. Narasimhan, P. Kar, and P. Jain. Optimizing Non-decomposable Performance Measures: A Tale of Two Classes. In *International Conference on Machine Learning*, ICML'15, pages 199–208, Lille, France, 2015. JMLR.org.

[78] N. Natarajan, I. S. Dhillon, P. D. Ravikumar, and A. Tewari. Learning with Noisy Labels. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[79] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. Rubinstein, U. Saini, C. A. Sutton, J. D. Tygar, and K. Xia. Exploiting Machine Learning to Subvert Your Spam Filter. In *First USENIX Workshop on Large-scale Exploits and Emergent Threats (LEET'08)*, 2008.

[80] B. Nelson, B. I. Rubinstein, L. Huang, A. D. Joseph, S. J. Lee, S. Rao, and J. Tygar. Query strategies for evading convex-inducing classifiers. *Journal of Machine Learning Research*, 13(May):1293–1332, 2012.

[81] L. Newman. How Malware Keeps Sneaking Past Google Play's Defenses. *Wired*, Sept. 2017.

[82] J. Newsome, B. Karp, and D. Song. Paragraph: Thwarting Signature Learning by Training Maliciously. In *Proceedings of the 9th International Conference on Recent Advances in Intrusion Detection*, RAID'06, pages 81–105, Berlin, Heidelberg, 2006. Springer-Verlag.

[83] A. Nguyen, J. Yosinski, and J. Clune. Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *CVPR*, pages 427–436, 2015.

[84] N. Papernot, P. McDaniel, and I. Goodfellow. Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples. *eprint arXiv:1605.07277*, 2016.

[85] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical Black-Box Attacks against Deep Learning Systems using Adversarial Examples. *eprint arXiv:1602.02697*, 2016.

[86] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The Limitations of Deep Learning in Adversarial Settings. In *European Symposium on Security & Privacy*, pages 372–387, 2016.

[87] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. *eprint arXiv:1511.04508*, 2015.

[88] B. I. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S.-h. Lau, S. Rao, N. Taft, and J. Tygar. ANTIDOTE: Understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, pages 1–14. ACM, 2009.

[89] S. Russell, D. Dewey, and M. Tegmark. Research priorities for robust and beneficial artificial intelligence. *Ai Magazine*, 36(4):105–114, 2015.

[90] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani. Software-Defined Networking for RSU Clouds in Support of the Internet of Vehicles. *IEEE Internet of Things Journal*, 2(2):133–144, Apr. 2015.

[91] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani. Reinforcement learning for resource provisioning in the vehicular cloud. *IEEE Wireless Communications*, 23(4):128–135, Aug. 2016.

[92] S. Sengupta, T. Chakraborti, and S. Kambhampati. Securing Deep Neural Nets against Adversarial Attacks with Moving Target Defense. *eprint arXiv:1705.07213*, May 2017.

[93] D. Silver. UCL Course on RL. http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html, 2015.

[94] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy (SP)*, pages 305–316. IEEE, 2010.

[95] J. Steinhardt, P. W. Koh, and P. Liang. Certified Defenses for Data Poisoning Attacks. *eprint arXiv:1706.03691*, June 2017.

[96] N. Suematsu and A. Hayashi. A Multiagent Reinforcement Learning Algorithm Using Extended Optimal Response. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1*, AAMAS '02, pages 370–377, New York, NY, USA, 2002. ACM.

[97] R. S. Sutton. Integrated Architecture for Learning, Planning, and Reacting Based on Approximating Dynamic Programming. In *Proceedings of the Seventh International Conference (1990) on Machine Learning*, pages 216–224, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[98] R. S. Sutton. Planning by Incremental Dynamic Programming. In *Proceedings of the Eighth International Conference on Machine Learning*, ML'91, pages 353–357, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.

[99] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, first edition, 1998.

[100] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing Properties of Neural Networks. *eprint arXiv:1312.6199*, 2013.

[101] F. Tramèr, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel. Ensemble Adversarial Training: Attacks and Defenses. *eprint arXiv:1705.07204*, May 2017.

[102] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. The Space of Transferable Adversarial Examples. *eprint arXiv:1704.03453*, 2017.

[103] J. Tukey. Mathematics and the Picturing of Data. In *Proc. Int. Congress of Mathematicians*, volume 2, pages 523–531, 1975.

[104] J. W. Tukey. The future of data analysis. *Annals of Matthematical Statistics*, 33:1–67, 1962.

[105] H. van Hasselt, A. Guez, and D. Silver. Deep Reinforcement Learning with Double Q-learning. *eprint arXiv:1509.06461*, Sept. 2015.

[106] M. J. Wainwright and M. I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, 2008.

[107] B. Wang, J. Gao, and Y. Qi. A Theoretical Framework for Robustness of (Deep) Classifiers against Adversarial Examples. *eprint arXiv:1612.00334*, 2016.

[108] H. Wang, W. Xing, K. Asif, and B. D. Ziebart. Adversarial Prediction Games for Multivariate Losses. In *Advances in Neural Information Processing Systems*, NIPS'15, pages 2728–2736, Cambridge, MA, USA, 2015. MIT Press.

[109] Z. Wang, C. Chen, H. X. Li, D. Dong, and T. J. Tarn. A novel incremental learning scheme for reinforcement learning in dynamic environments. In *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pages 2426–2431, June 2016.

[110] M. Weinberg and J. S. Rosenschein. Best-Response Multiagent Learning in Non-Stationary Environments. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '04, pages 506–513, Washington, DC, USA, 2004. IEEE Computer Society.

[111] S. a. E. Wikipedia. Sturmovik at english: Vandalized stop sign at the Brunswick MARC station in Brunswick Maryland. Digitally edited, Jan. 2008.

[112] H. Xiao, H. Xiao, and C. Eckert. Adversarial Label Flips Attack on Support Vector Machines. In *Proceedings of the 20th European Conference on Artificial Intelligence*, ECAI'12, pages 870–875, Amsterdam, The Netherlands, The Netherlands, 2012. IOS Press.

[113] W. Xu, D. Evans, and Y. Qi. Feature Squeezing: Detecting Adversarial Examples in Deep Neural Networks. *eprint arXiv:1704.01155*, 2017.

[114] D. Ye, M. Zhang, and D. Sutanto. A Hybrid Multiagent Framework With Q-Learning for Power Grid Systems Restoration. *IEEE Transactions on Power Systems*, 26(4):2434–2441, Nov. 2011.

[115] K. Zetter. A Google Site Meant to Protect You is Helping Hackers Attack You. *Wired*, Aug.

2014.

[116] F. Zhang, P. P. K. Chan, B. Biggio, D. S. Yeung, and F. Roli. Adversarial Feature Selection Against Evasion Attacks. *IEEE Transactions on Cybernetics*, 46(3):766–777, Mar. 2016.

[117] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the Robustness of Deep Neural Networks via Stability Training. *eprint arXiv:1604.04326*, 2016.

[118] S. Zhifei and E. Meng Joo. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3):293–311, 2012.

Table 1: Taxonomy of attacks on supervised machine learners, with representative past work. Attacks against reinforcement learning are covered separately in Section 4.2.2.

		Integrity	**Availability**
Causative	**Targeted** (training set manipulation for specific errors)	Rubinstein et al. [88]: boiling frog attacks against the PCA anomaly detection algorithm; Li et al. [62]: poison training data against collaborative filtering systems; Mei & Zhu [69]: identify the optimal training set to manipulate different machine learners; Burkard & Lagesse [28]: targeted causative attack on Support Vector Machines that are learning from a data stream.	Newsome et al. [82]: manipulate training set of classifiers for worms and spam to block legitimate instances; Chung & Mok [33]: generate harmful signatures to filter out legitimate network traffic; Nelson et al. [79]: exploit statistical machine learning against a popular email spam filter.
	Indiscriminate (training set manipulation to maximise overall error rate)	Biggio et al. [24]: inject crafted training data to increase SVM's test error; Xiao et al. [112]: label flips attack against SVMs; Koh & Liang [56]: minimise the number of crafted training data via influence analysis.	Newsome et al. [82]; Chung & Mok [33]; Nelson et al. [79].
Exploratory	**Targeted** (test instance manipulation for specific errors)	Nelson et al. [80]: probe a classifier to determine good attack points; Papernot et al. [86]: exploits forward derivatives to search for the minimum regions of the inputs to perturb; Goodfellow et al. [43]: design the "fast gradient sign method" (FGSM) to generate adversarial samples; Carlini & Wagner [31]: propose the C&W method for creating adversarial samples; Han & Rubinstein [45]: improve the gradient descent method by replacing with gradient quotient.	Moore et al. [74]: provide quantitative estimates of denial-of-service activity.
	Indiscriminate (test instance manipulation for misclassification)	Biggio et al. [23]: use gradient descent method to find attack instances against SVMs; Szegedy et al. [100] demonstrate that changes imperceptible to human eyes can make DNNs misclassify an image; Goodfellow et al. [43]; Papernot et al. [85, 84]: attack the target learner via a surrogate model; Moosavi-Dezfooli et al. [75, 76]: propose DeepFool that generates universal perturbations to fool multiple DNNs; Carlini & Wagner [31]; Nguyen et al. [83]: produce images that are unrecognisable to humans, but can be recognised by DNNs; Han & Rubinstein [45].	Moore et al. [74].

Table 2: Details of the more complex SDN example (see Figure 6).

	Defender	**Attacker**
State	(1) Whether each node is compromised; (2) Whether each link is turned on/off.	
Actions	(1) Configure anomaly detection at a node; (2) Isolate a node and re-routing its links; (3) Reconnect a node and its links; (4) Migrate the critical server and select the destination.	Compromise a node that satisfies certain conditions, e.g. if the node a) is closer to the “backbone" network; b) is in the backbone network; or c) in the target subset.
Goals	(1) Protect as many nodes as possible; (2) Turn off as few links as possible; (3) Isolate the compromised nodes; (4) Ensure the critical node can’t be compromised; (5) Ensure the links connecting it with the rest of the network can’t be cut off.	Compromise the target server.

Table 3: NGTF AMLC Project timetable. CY ≡ Calendar Year.

		CY1-H2	CY2-H1	CY2-H2	CY3-H1	CY3-H2	CY4-H1
Scoping	Study						
Pillar 1	WP 1.1.1						
	WP 1.1.2						
	WP 1.1.3						
Pillar 2	WP 2.1.1						
	WP 2.2.1						
	WP 2.2.2						
	WP 2.3.1						
	WP 2.3.2						
	WP 2.3.3						
	WP 2.4.1						
	WP 2.4.2						
Pillar 3	WP 3.1.1						
	WP 3.2.1						
	WP 3.2.2						
	WP 3.3.1						
	WP 3.3.2						
	WP 3.4.1						
	WP 3.4.2						

This page is intentionally blank

DISTRIBUTION LIST

Adversarial Machine Learning
for Cyber-Security:

NGTF Project Scoping Study

AMLC Team at UniMelb, Data61 and Swinburne Univ.
Tamas Abraham, Olivier de Vel, Paul Montague

S&T Program

Chief of Cyber and Electronic Warfare Division	1
Authors: DST Group Authors (Abraham T., de Vel O. and Montague P.)	1

DEFENCE SCIENCE AND TECHNOLOGY GROUP DOCUMENT CONTROL DATA			1. DLM/CAVEAT (OF DOCUMENT)
2. TITLE Adversarial Machine Learning for Cyber Security: NGTF Project Scoping Study		3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED LIMITED RELEASE USE (U/L) NEXT TO DOCUMENT CLASSIFICATION) Document (U) Title (U) Abstract (U)	
4. AUTHOR(S) AMLC Team at UniMelb, Data61 and Swinburne Univ. Tamas Abraham , Olivier de Vel, Paul Montague		5. CORPORATE AUTHOR Defence Science and Technology Group PO Box 1500 Edinburgh, South Australia 5111	
6a. DST GROUP NUMBER DST-Group-GD-0988	6b. AR NUMBER AR-017-073	6c. TYPE OF REPORT General Document	7. DOCUMENT DATE January 2018
8. OBJECTIVE ID	9.TASK NUMBER N/A	10.TASK SPONSOR N/A	
11. MSTC N/A		12. STC N/A	
13. DOWNGRADING/DELIMITING INSTRUCTIONS		14. RELEASE AUTHORITY Chief, Cyber and Electronic Warfare Division	
15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT *Approved for public release.* OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111			
16. DELIBERATE ANNOUNCEMENT No limitations			
17. CITATION IN OTHER DOCUMENTS Yes			
18. RESEARCH LIBRARY THESAURUS Adversarial machine learning, software-defined network, cyber-security, autonomy			
19. ABSTRACT This report is the result of a scoping study undertaken as part of an Australian Department of Defence Next Generation Technologies Fund (NGTF) project entitled Adversarial Machine Learning for Cyber- Security (AMLC). The report describes the broader context for the project (e.g. attacks and defences against machine learners), outlines general concepts and techniques for adversarial machine learning, and focusses on reinforcement machine learning algorithms of specific relevance to Defence. A software simulation platform will also be developed to demonstrate the effectiveness of the attacks against, and defences of, such machine learning algorithms in a cyber-security context.			