Empirical Mixnet Design

Yongzhao Wang[1], Tariq Elahi[2], Vasilios Mavroudis[3], Edward Plumb[4], Rahul Savani[5], and Theodore Turocy[6]

[1] University of North Florida, yongzhao.w@unf.com
[2] University of Edinburgh, t.elahi@ed.ac.uk
[3] The Alan Turing Institute, vmavroudis@turing.com
[4] London School of Economics and Political Science, e.plumb@lse.ac.uk
[5] University of Liverpool, rahul.savani@liverpool.ac.uk
[6] University of East Anglia, T.Turocy@uea.ac.uk

Abstract. Mixing networks (*mixnets*) are cryptographic protocols designed to enhance the privacy and anonymity of online communications. Despite their robust privacy features, mixnets remain vulnerable to cyber attacks, such as adversaries compromising mixing nodes (*mixes*) to track message flows. To mitigate these threats, we propose an empirical mechanism design framework for identifying mixnet configurations that satisfy the designer's goals. We first introduce two models for mixnets: one with heterogeneous mixes and another with layer-wise homogeneous mixes, capturing the diverse scales and complexities of real-world mixnets. The heterogeneous model allows each mix to have different properties, offering greater expressiveness but posing challenges for scalability of analysis. In contrast, the homogeneous model provides a simplified, yet practical, framework for large-scale analysis. Leveraging these models, we conceptualize adversarial attacks and defensive measures as strategic interactions between an attacker and a defender in a *game*. We demonstrate how our framework generates data that is useful for a mechanism designer trading off cost versus performance in deploying a mixnet.

Keywords: Mixnet · Empirical Mechanism Design · Policy Space Response Oracles.

1 Introduction

Mixing networks, or ***mixnets***, first proposed by Chaum [1981], are cryptographic protocols designed to provide privacy and anonymity in online communication and data transmission. They operate by shuffling messages and routing them independently through a series of intermediary servers called ***mixes*** or ***mixing nodes***, making it extremely difficult for *passive* adversaries[7] to trace the origin or destination of any given message. Due to its resistance to passive adversaries, mixnets become a critical component of privacy-preserving technologies and have been widely employed in modern communication systems, such as the Nym network [Diaz *et al.*, 2021], the XX network [xx Network, 2021], and HOPR [HOPR, 2021; Benedicic *et al.*, 2022].

[7] Passive adversaries are entities that monitor but do not interfere with network communications.

Despite their robustness against passive adversaries, sophisticated *active* adversaries, henceforth attackers, can still infiltrate a mixnet and undermine its anonymity. For example, attackers may compromise existing mixes or set up their own mixes to control end-to-end paths and trace users. In the most severe scenarios, attackers could fully observe the links within a mixnet, uncovering its mixing strategies and infrastructure. To counteract these attacks, defenders need to scrutinize the behavior of mixes to identify potential threats and expel suspicious mixes from the network while introducing mixes from trusted entities. The interplay between attackers and defenders, each striving to manipulate the mixnet to serve their objectives, inherently constitutes a *game*.

This study aims to achieve two main goals: first, to model the dynamics of this interplay using game theory, and second, to demonstrate how the resulting models generate data that can be useful for a mechanism designer trading off cost against performance when deciding on a mixnet to deploy. To model the dynamics, we first introduce ***heterogeneous mixnet games***, which use a stratified mixnet topology (i.e., with *layers* of mixing nodes) and assign distinct attributes to each mix in the mixnet. The heterogeneity offers the greatest expressiveness but poses challenges for scalability of analysis. For scalability, we further present a simplified, yet practical, version called ***homogeneous mixnet games***. In a homogeneous mixnet game, the underlying mixnet includes *homogeneous* mixes *within each layer* of a stratified topology, but allows *heterogeneity* of mixing node properties *across* layers. Compared to the heterogeneous model, this simplification sacrifices some of the finer details of mixnets, while being practical for large-scale mixnet analysis.

Both mixnet games model the interaction between two strategic centralized players, an attacker and a defender, who compete for control over mixes in the underlying mixnet. The attacker can either deploy its own mixes or compromise existing honest ones, aiming to gain control over paths through the mixnet and thereby monitor user traffic. In response, the defender can deploy uncompromised mixes or exclude potentially compromised ones from the network, relying on noisy alarm signals. Both players incur costs for their actions. In addition, rewards and penalties are assigned when user communication paths fall under the attacker's control. The realized costs, rewards, and penalties are determined by the actions of both players and the particular configuration of the mixnet (e.g., the security level of mixes), collectively shaping the utility (i.e., payoff) functions of the mixnet games.

We then consider an ***empirical mechanism design*** [Vorobeychik *et al.*, 2006] problem for mixnets, where a designer can control certain characteristics of the mixnet to affect the rules of mixnet games (i.e., the mechanism) and achieve its design goals. Because the performance of a mixnet is mediated by the strategic behavior of the players, this behavior must be incorporated in the analysis. Mechanism design customarily adopts ***Nash equilibrium*** (NE) as the prediction for how players will play the game. However, due to the vast state and action spaces in the mixnet game, equilibrium computation methods are computationally impractical. To tackle this challenge, we utilize ***Policy Space Response Oracles*** (PSRO) [Lanctot *et al.*, 2017; Bighashdel *et al.*, 2024], an iterative method for approximating NE in large games using a combination of deep reinforcement learning (DRL) and traditional equilibrium computation methods. While

PSRO has been previously applied in analyzing a single game, our work extends it to empirical mechanism design, where PSRO is used to analyze multiple games (i.e., mechanism candidates).

We demonstrate how a mechanism designer can use our models for design purposes. Specifically, we assume that the designer can choose from configurations with mixes of varying security levels. High-security mixes offer stronger protection against attacks but incur greater maintenance costs for the defender, while low-security mixes are less costly but more vulnerable to compromise. Through an equilibrium analysis using PSRO across different configurations, we show the tradeoffs between the defender's cost and the mixnet performance. We observe that the number of compromised paths decreases monotonically as the mixnet's security level increases. Furthermore, we find that different components of defense costs respond differently to increasing attack intensity, revealing a fundamental tradeoff. These insights allow mixnet designers to systematically assess these tradeoffs and choose configurations that best balance security, cost, and performance requirements.

This work makes the following key contributions:

1. **Two novel mixnet models**: A *heterogeneous model* and a *homogeneous model*, capturing distinct granularities of mixnet architectures;
2. **Game-theoretic formalization**: A competitive framework between attackers and defenders, including player roles, strategies, and utility functions;
3. **Equilibrium analysis**: Application of PSRO to empirical mechanism design for evaluating multiple mechanism candidates;
4. **Cost-performance tradeoffs**: Insights for mechanism designers to optimize system design by trading off defense costs against mixnet performance.

2 Related Work

2.1 Mixnets

Mixnets play a crucial role in enhancing privacy by ensuring unlinkability through layered encryption and randomized mixing. Early mixnet designs, like those introduced by Chaum [1981], provide robust privacy guarantees based on computational security, but they can be resource-intensive and challenging to implement in large-scale, real-world scenarios. Consequently, most modern mixnet designs focus on finding a balance between security and efficiency, using topological and operational defenses that account for realistic adversarial models. In this section, we discuss these approaches.

Network Topology. Several studies have investigated the design of mixnet topologies to strengthen their security. For example, the Nym mixnet [Diaz *et al.*, 2021] employed a stratified topology, periodically randomizing mix placement to thwart adversaries from strategically positioning their nodes for maximum influence at minimal cost. However, this randomization may result in inefficient resource utilization, as mix capacities are not considered in the process. To overcome this issue, Ma *et al.* [2022] introduced a solution involving guard mixes and randomization, enabling both optimal resource allocation and privacy protection. Another strategy, called location-aware mix selection

[Rahimi *et al.*, 2024], seeks to diversify the network by choosing mixes based on geographic distribution, thereby distributing trust more effectively.

Parameterization. Mixnet designs are often parameterized to facilitate trade-offs between privacy and performance, making them adaptable to different deployment scenarios. One significant parameter, relevant to this work, is the number of layers in the network's stratified topology. Empirical studies have demonstrated that a three-layer stratified topology provides an effective compromise between privacy and performance [Ben Guirat *et al.*, 2021; Piotrowska *et al.*, 2017; Guirat and Diaz, 2022]. These studies primarily involve manually searching the parameter space under a static adversarial model, where compromised nodes are selected at the outset and remain fixed.

Deployment. Deployment strategies are vital for ensuring both the security and efficiency of mix networks. Game-theoretic methods have been used to optimize the deployment of Tor pluggable transports [Elahi *et al.*, 2016], focusing on how many transports to deploy at once and how best to distribute traffic among them to avoid censorship. Moreover, Soleimani *et al.* [2018] applied game theory to model the challenge of distributing Tor proxy IP addresses as a strategic interaction between system operators and censors. Additionally, research like the incentive schemes proposed by Halpin and Serjantov [2023] sought to motivate participants to behave honestly, further strengthening the privacy infrastructure.

2.2 Empirical Mechanism Design

Mechanism design involves creating a set of rules that govern interactions among players to achieve specific design goals. The mechanism, combined with the players' preferences, defines a game and thus the mechanism designer is essentially setting up the game. To assess a mechanism, one must solve this game and evaluate its outcomes. With such an evaluation in place, the designer can, in theory, search for mechanisms that optimize their desired outcomes. Our work falls within the domain of empirical mechanism design (EMD) [Vorobeychik *et al.*, 2006], a branch of mechanism design that leverages real-world or simulated data instead of relying solely on theoretical models and assumptions. While traditional mechanism design typically addresses private information such as players' types, EMD allows for design under uncertainty about behavioral parameters, aligning our approach with this literature.

For EMD, Vorobeychik *et al.* [2006] applied EMD to the Trading Agent Competition (TAC) supply chain management tournament, using Empirical Game-Theoretic Analysis [Wellman, 2006] to evaluate design adjustments, such as altering storage costs to reduce excessive procurement. Similarly, Jordan *et al.* [2010] analyzed TAC Ad Auctions, studying how auction parameters like reserve price influenced publisher revenue by re-equilibrating gameplay using empirical game models. Brinkman and Wellman [2017] also explored optimal clearing intervals for call markets, while Gatchel and Wiedenbeck [2023] proposed a learning method to infer utility functions for mechanism design objectives.

In contrast to evaluating fixed mechanism candidates, Vorobeychik *et al.* [2012] and Areyan Viqueira *et al.* [2019] introduced black-box optimization, using stochastic search and Bayesian optimization to identify revenue-maximizing mechanisms.

3 Mixnet Structures

A mixnet is given by a directed graph $\mathcal{G} = (V, E)$, where vertices $v \in V = V_u \bigcup V_m$ represent users V_u and potential positions of mixes V_m, and edges $e \in E$ are communication links. We consider a typical stratified topology for the mixnet where mixes are organized in layers, as shown in Figure 1, and we assume mixes within two consecutive layers are fully connected. In particular, user nodes (orange nodes) are located on both sides of the graph and one side is a mirror of another, representing the same group of users. Potential positions for the deployment of mixes V_m are organized in the middle layers. Note that due to full connectivity between layers, the specific position of a mix within a layer is irrelevant; the positions shown are for illustrative purposes only.

A ***user-user path*** in the mixnet is a sequence of connected edges across layers, starting at a sender (user) and ending at a receiver (another user). Formally, a user-user path $p = (v_0, v_1, \ldots, v_k)$ is a sequence, where $v_0 \in V_u$, $v_k \in V_u$, and $v_i, v_{i+1} \in V_m$ are in two consecutive layers for $0 < i < k - 1$. A ***communication path*** is a directed path that excludes the sender and receiver, denoted as $p_c = (v_1, \ldots, v_{k-1})$, where each position v_i has a mix deployed. The ***bandwidth*** of the mixnet is the total number of distinct communication paths in the mixnet. In this work, we focus on the portion of the mixnet concerning the deployment of mixes (i.e., V_m), while disregarding the users' nodes, and, henceforth, we refer to communication paths simply as paths. Finally, we assume that there is a sufficient amount of traffic, including cover traffic, in the mixnet to preclude the possibility of traffic analysis-based attacks.

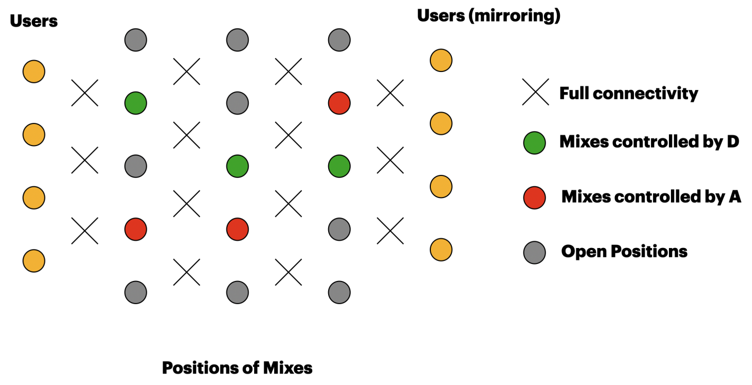


Fig. 1: Illustration of a stratified topology of a mixnet. Green mixes represent honest mixes (controlled by the defender), while red mixes are compromised mixes (controlled by the attacker). Grey mixes are open positions where new mixes can be deployed. Together, these mixes constitute the set of potential mix positions, denoted as V_m.

To accommodate the varying scales and complexities of real-world mixnets, we introduce two mixnet models: one with heterogeneous mixes and another with layer-wise homogeneous mixes. The heterogeneous model permits each mix to have different inherent properties, for example, maintenance cost, likelihood to be compromised under an attack, false alarm rate, and so on. While heterogeneity can enhance expressiveness, it also complicates the analysis of a mixnet, as each individual mix must be analyzed independently, making scalability difficult. To address this, we propose a homogeneous model, where mixes within each layer share the same inherent properties, while heterogeneity is allowed across layers. This design resembles large-scale mixnets in the real world and supports their analysis by focusing on the proportion of mixes controlled by players, rather than on specific mixes.

4 Mixnet Games

Building on these two mixnet models, we define ***mixnet games*** between an attacker and a defender. To surveil as many users as possible, the attacker seeks to maximize the number of fully compromised paths (those consisting solely of mixes under its control) by either deploying its own malicious mixes or compromising existing honest ones. In defense, the defender can mitigate adversarial influence by deploying trusted mixes or excluding suspected compromised nodes. Our mixnet games rely on the *any trust* assumption, which requires that *at least one mix* on a communication path be *honest* to ensure privacy [Wolinsky *et al.*, 2012a,b; Corrigan-Gibbs *et al.*, 2013; Ma *et al.*, 2022; Lazar and Zeldovich, 2016]. In the security literature, these together define a ***threat model***, which outlines the system's security assumptions, adversary capabilities, and potential countermeasures.

4.1 The General Heterogeneous Mixnet Game

We model the strategic interaction between an attacker and a defender as a ***(heterogeneous) mixnet game***. We assume that the attacker and the defender are centralized players who can respectively launch attacks and deploy defenses across the mixnet in a coordinated fashion.

Fix a mixnet, with a given set of potential mix positions V_m, and a given topology. Each position $v \in V_m$ has a state, which is given by $z(v) \in Z \equiv \{\omega, 0, 1\}$, where $z(v) = 0$ is a mix initially controlled by the defender, $z(v) = 1$ a mix controlled by the attacker, and $z(v) = \omega$ is an open position without a mix. The attacker observes the state of the mixnet perfectly. However, the defender only receives a noisy observation $o_d(v)$ of the state of each position v. The defender knows which positions do not have mixes currently deployed, so $o_d(v) = \omega$ whenever $z(v) = \omega$. When $z(v) = 0$, the defender receives a ***false alarm*** $o_d(v) = 1$ with probability $\alpha \in [0, 1]$; otherwise they observe $o_d(v) = 0$. When $z(v) = 1$, the defender receives a ***false negative*** $o_d(v) = 0$ with probability $\beta \in [0, 1]$; otherwise they observe $o_d(v) = 1$. The observations on each position are realized independently.

We organize the interaction between the attacker and the defender into ***episodes*** of T time steps of the game, starting from an ***initial state*** $z_0(v)$ for each position $v \in V_m$ at $t = 0$. At each time step $t < T$, both the attacker and the defender choose an action for each position v. Their actions and the current state $z_t(v)$ determine the next state $z_{t+1}(v)$ of the position at $t + 1$. For open positions with $z_t(v) = \omega$, either player may choose to *deploy* or *not deploy* a mix. Deploying a mix at node v incurs a cost of $c_d(v)$. If the defender deploys at v, then the position is controlled by the defender, $z_{t+1}(v) = 0$. If the defender does not deploy but the attacker does, then the position is controlled by the attacker, $z_{t+1}(v) = 1$. If neither deploy, then $z_{t+1}(v) = \omega$. For positions with $z_t(v) \neq \omega$, the defender may choose to *exclude* or *not exclude* the position. If the defender chooses to exclude, they pay a cost of $c_e(v)$, and then $z_{t+1}(v) = \omega$, irrespective of what the attacker does at that position. Otherwise, for positions with $z_t(v) = 0$, the attacker may choose to *attack* or *not attack*. If the attacker attacks, they incur a cost $c_a(v)$. The attack succeeds with probability $\rho(v) \in (0, 1]$. A successful attack results in $z_{t+1}(v) = 1$; otherwise $z_{t+1}(v) = 0$. At any position where the defender does not exclude and the attacker does not attack, then $z_{t+1}(v) = z_t(v)$.

A ***pure strategy*** for each player consists of the choice of an action at each position given its observation. Let S_d denote the set of strategies for the defender, and S_a the set of strategies for the attacker. The objectives of the defender and attacker are captured by their ***utility functions***. We denote the defender's utility function as $u_d(s_a, s_d)$. The defender's utility incorporates five elements:

– **Exclusion costs c_e**: Excluding an existing mix incurs a cost;
– **Deployment costs c_d**: Deploying a new mix incurs a cost;
– **Maintenance costs c_m**: The defender incurs ongoing costs for each active mix they have deployed previously (regardless of compromise status), reflecting the expense of holding these mixes;
– **Penalty for compromised paths**: The defender is penalized proportionally to the number of fully compromised paths;
– **Penalty for low bandwidth**: If the total number of paths falls below a specified threshold (regardless of compromise status), the defender incurs a penalty due to insufficient communication bandwidth for user traffic.

In particular, we refer to the sum of exclusion costs, deployment costs, and maintenance costs as the ***defense costs***.

The attacker's utility function $u_a(s_a, s_d)$ incorporates four elements:

– **Attack costs c_a**: The cost of attacking an existing mix;
– **Deployment costs c'_d**: Deploying a new mix incurs a cost, regardless of its success (defender deployments always succeed, while attacker deployments may be overridden);
– **Maintenance costs c'_m**: The attacker incurs ongoing costs for each mix they have deployed previously;
– **Reward for compromised paths**: The attacker is rewarded proportionally to the number of fully compromised paths.

Note that each parameter can be specified separately for each position in V_m, and the defender's deployment and maintenance costs can differ from those of the attacker. Because the defender's observation of the state may be stochastic, utilities are defined formally as expectations across observations. The utility function for an episode is computed as the expected value of the sum of utilities across the time steps in the episode. The strategy sets and utility functions as described above form a *two-person strategic-form game*.

4.2 The Homogeneous Mixnet Game

For even modestly-sized mixnets, the strategy spaces in the general mixnet game are enormous. There are $3^{|V_m|}$ possible states (or observations of states), and in most cases a player has two actions they could take at each position, leading to up to $2^{3^{|V_m|}}$ strategies.

Our principal technique for addressing this complexity is via the use of PSRO to consider only a small subset of strategies, as described in more detail in the next section. However, the full generality of the model, allowing security levels and maintenance costs to be specified on a per-node basis, is not essential for generating insights about design tradeoffs. We therefore define a ***homogeneous mixnet game***, which places more structure on the parameters of the mixnet, and restricts the strategy spaces of the players.

In the homogeneous model, all mixes within a given layer share identical inherent properties, while allowing heterogeneity between layers. Exploiting this symmetry, we treat individual positions within each layer as anonymous. We assume that instead of observing the entire state of each layer, the attacker conditions their action based only on the *proportions* of positions in the layer that are in each of the three states. Likewise, the defender's observation consists only of the aggregated signals, generating an observed proportion of positions at which alarms are and are not triggered. This is done by summing up the signals generated as in the full heterogeneous model.

The actions of the attacker and the defender are specified in an aggregate fashion for each layer. The attacker chooses how many of the open positions at which to deploy mixes, and how many of the positions controlled by the defender to attack. Similarly, the defender chooses how many of the open positions at which to deploy mixes, and how many of the positions at which they receive an alarm to remove from the mix. The actions of both players are implemented by randomly choosing the positions to which to apply the action to, after which the end state of the mixnet and the players' payoffs are determined as in the full heterogeneous case. That is to say, in the homogeneous model we restrict attention only to a form of symmetric strategies, in which any two positions which are symmetric with respect to the mixnet are treated symmetrically in the players' strategies.

5 Methods

5.1 Empirical Mechanism Design

We consider the problem of a designer who is able to control certain characteristics of the network. For our exercise, we assume that the designer is able to affect the attack success probability by setting the *security levels* of positions, thereby determining the security levels of mixes to be deployed at those positions. Specifically, the security level at a position $v \in V_m$ affects the probability $\rho(v)$ that an attack at position v is successful, and also the cost of maintenance $c_m(v)$ of the defender maintaining a mix at that position.

Mechanism design often involves considering multiple criteria for evaluating the performance of a candidate solution. For our analysis, we assume that the designer ultimately cares about two measures of performance. First, the designer aims to optimize the mixnet performance by minimizing the number of compromised paths. Second, the designer seeks to reduce the defender's costs incurred both when investing in higher security levels during the design phase and when defending the mixnet after deployment. Therefore, the designer faces a cost-performance tradeoff between those design-time fixed costs, along with running-time incremental costs, and overall performance of the mixnet.

In general, designers will face some tradeoffs among the criteria of interest. This can be formalized by defining an objective function for the designer, and then optimizing with respect to that objective function. For the purpose of our exercise, we will remain agnostic as to the form of this objective function, and instead focus on illustrating the nature of the tradeoffs a mixnet designer would face.

The link between the choice variables of the designer and the performance of the mixnet is mediated by the strategic responses of the attacker and the defender in the mixnet game. In particular, in mechanism design the designer knows that the players in the game will have access to information that the designer does not. For example, in our setting, the players will know the actual values of the various costs of actions, while the designer only knows they will fall in some range.

5.2 Modeling Player Behavior via PSRO

The modeling of these strategic responses is the principal challenge faced by the designer. First, it is straightforward to see that in the mixnet game, both players have incentives to be unpredictable in choosing the positions at which they deploy mixes, execute attacks, or remove from the mix. This is captured by allowing the attacker and defender to play ***mixed strategies*** σ_a and σ_d, which are probability distributions over their finite strategy sets.

In mechanism design, the concept of Nash equilibrium is often used to specify the behavior of players. An NE is a mixed strategy profile $\sigma = (\sigma_a, \sigma_d)$ such that neither player can improve their expected payoff by unilaterally deviating from their strategy. Formally, σ is an NE if and only if $U_a(s_a, \sigma_d) \leq U_a(\sigma_a, \sigma_d)$ for all $s_a \in S_a$ and $U_d(\sigma_a, s_d) \leq U_d(\sigma_a, \sigma_d)$ for all $s_d \in S_d$. However, given the vast state and action spaces in a mixnet game, it is not computationally feasible to find Nash equilibrium strategy profiles. Instead, we compute a prediction of how players will play in a given mixnet game using PSRO. The PSRO algorithm provides an iterative framework for exploring the strategy space and approximating NE in large games.

We illustrate a simplified version of PSRO in Figure 2. PSRO starts from a set of initial strategies for each player, forming the initial ***restricted game***. The expected payoffs of strategy profiles in the restricted game are estimated through simulation. At each iteration of PSRO, an NE of the restricted game is computed and each player finds their best response to the equilibrium strategies of the other players using deep reinforcement learning. These newly identified best responses (i.e., neural-network strategies) are then added to the restricted game, expanding the restricted strategy space. This process continues until no player can improve by deviating from the NE of the restricted game using DRL. At this point, the NE of the restricted game becomes an approximate NE of the full game (the approximation arises from estimating payoffs via simulation and computing best responses using DRL).

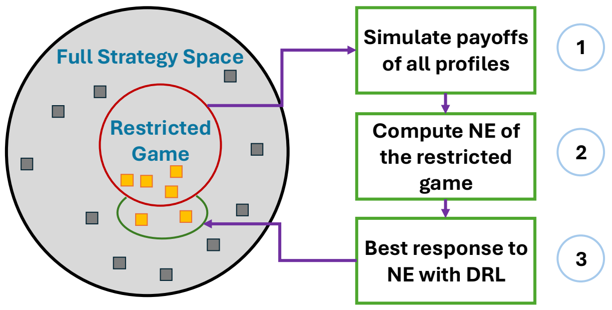

Full Strategy Space
Restricted Game
Simulate payoffs of all profiles
1
Compute NE of the restricted game
2
Best response to NE with DRL
3

Fig. 2: An illustration of PSRO. PSRO iteratively constructs a restricted game with the purpose of capturing a good approximate NE

For a neural-network strategy, we employ the network architecture depicted in Figure 3. The input to this neural network is the player's observation on mixes or layers of mixes at each time step, while the output is the player's atomic actions on these mixes or the layers, depending on the mixnet model and the player's role.

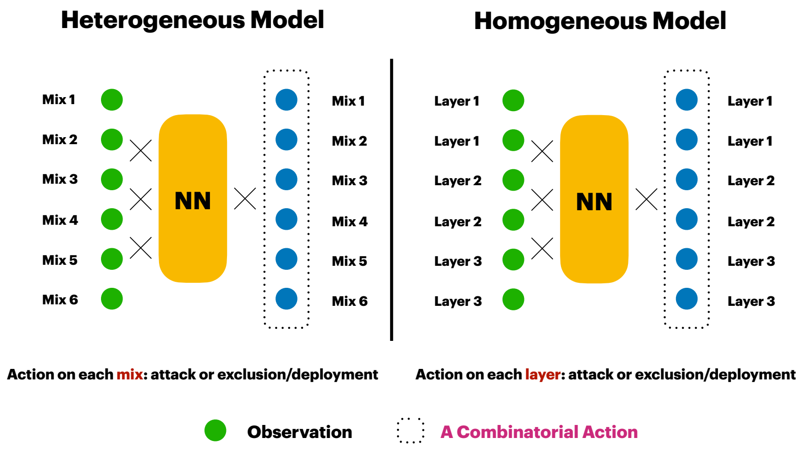


Fig. 3: An illustration of neural-network strategies.

Given an instance of the mixnet game, we apply PSRO to approximate NE. We employ a regularization technique to accelerate the PSRO convergence [Wang and Wellman, 2023]. After PSRO stops, an NE of the final restricted game is computed using Gambit [Savani and Turocy, 2024], in particular, using a pivoting method for linear complementarity problems[8]. The found NE of this final restricted game, which is a mixed strategy profile, is the output of PSRO, as in the approach of Li *et al.* [2023]. We refer to the mixed strategy profile resulting from this process as the ***solution*** of the mixnet game produced by PSRO, and use this profile as the input to calculations evaluating the performance of different mixnet designs.

5.3 Constructing Combinatorial Actions

In a mixnet game with the heterogeneous model, each player's overall action consists of a combination of atomic actions applied across all mixes, resulting in a combinatorial action space. Optimizing strategies over such a vast space is computationally infeasible. To address this, we adopt a ***sequential action selection*** approach. Instead of selecting from the full combinatorial set, sequential selection constructs an action by making a series of decisions over atomic actions, effectively avoiding the need to handle the full exponential complexity. One successful application of this approach can be found in AlphaStar [Vinyals *et al.*, 2017], which utilized it to decompose the joint distribution over actions into marginal distributions, subsequently diminishing the action's dimensionality through sequential action selection based on these marginal distributions. We illustrate the construction of a combinatorial action in Figure 4.

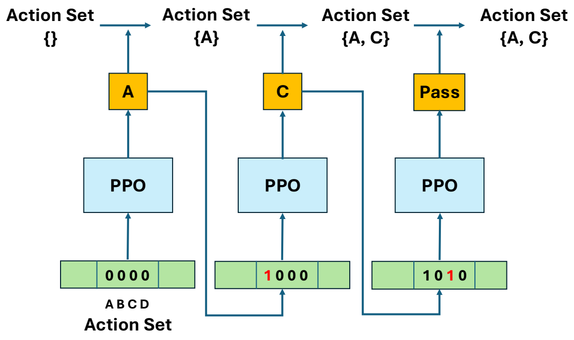


Fig. 4: Flowchart describing the autoregressive action set construction. The input includes the current atomic action set with 4 bits. Initially the bits are set to zeros to indicate that the action set is empty. Then the bits flip if corresponding atomic actions were added.

For the homogeneous model, we directly optimize over the combinatorial action space for training a strategy, as its dimensionality (determined by the number of layers) is small enough to make optimization feasible.

[8] In general-sum games, there may be multiple NE with different strategic behavior and correspondingly different mixnet performance.

6 Experiments

6.1 Experimental Setup

To illustrate how our EMD framework works and shows useful cost-performance trade-offs for mechanism design, we set up the following parameters for our experiments. In practice, a mechanism designer can plug in any real-world parameters and select sensible objective functions to optimize.

The key choice variable for the designer is the investment they make in the quality of the security of the mixes. This quality is measured by the probability of success $\rho(v)$ of an attack at that node, and the defender's cost $c_m(v)$ of maintaining a mix at that node. We consider four regimes of security levels:

- A: $\rho \in [0.2, 0.4]$ and $c_m \in [40, 50]$;
- B: $\rho \in [0.4, 0.6]$ and $c_m \in [30, 40]$;
- C: $\rho \in [0.6, 0.8]$ and $c_m \in [20, 30]$;
- D: $\rho \in [0.8, 1.0]$ and $c_m \in [10, 20]$.

In these regimes, high-security mixes reduce the likelihood of a successful attack, but incur greater maintenance expenses, while low-security mixes are more cost-effective to maintain but face a higher risk of being compromised. This stems from the reality that high-security mixes often demand more specialized personnel, continuous monitoring, prompt issue resolution, higher server running costs, and more frequent audits, among other requirements.

To create a heterogeneous mixnet game instance of a given security level, we first set up a mixnet consisting of three layers, each containing 10 positions. The aggregate initial state of the mixnet is set such that 80% of the mixes are honest, 10% are compromised, and 10% are open positions in each layer. The initial category assignments of the mixes are determined randomly. For each position v in the mixnet, we randomly generate its attributes as follows:

- The probability of successful attack $\rho(v)$ from the corresponding interval;
- The cost of maintenance $c_d(v)$ from the corresponding interval;
- A false alarm rate $\alpha(v) \in [0.2, 0.4]$;
- A false negative rate $\beta(v) \in [0.1, 0.3]$;
- A deployment cost $c_d(v) \in [10, 30]$;
- A attack cost $c_a(v) \in [30, 50]$;
- An exclusion cost $c_e(v) \in [30, 50]$.

We select the last five parameters at random to reflect that the designer does not know in advance the conditions that will pertain. We sample the first two parameters randomly from within specified ranges to avoid degenerate cases due to payoff ties or non-representative situations which occur due to a specific pattern of realized parameters.

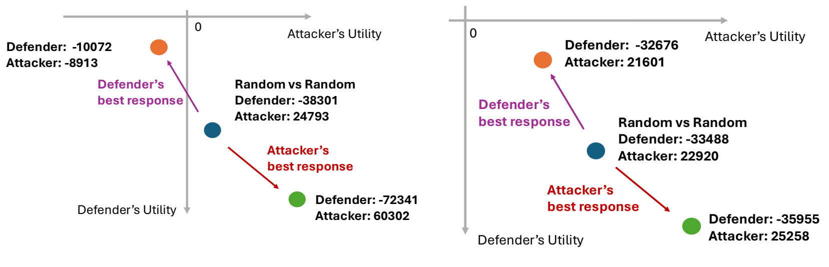
At each time step, the attacker receives a reward equal to 10 times the number of compromised paths, which corresponds to an equivalent penalty imposed on the defender. We require that the number of paths connecting users be at least 20% of the total possible paths—defined as the product of the number of nodes in each layer. If this threshold is not met, a substantial penalty is imposed on the defender to reflect the reduced communication bandwidth of the mixnet. This penalty incentivizes the defender to maintain a minimum level of communication bandwidth. The utilities of the attacker and defender are computed by summing up the separate components of their payoffs.

Experiments with the homogeneous model use similar distributions of parameter values with small adjustments. In keeping with the assumptions of the homogeneous model, all mixes in a given layer share the same realized probabilities and costs. Within the homogeneous instances, we set up three layers, with 1000 positions in each layer.

Once a mixnet game instance is created, we apply PSRO to approximate NE of that instance. For each episode of playing a strategy profile, we set the total number of time steps to $T = 10$ and use $K = 1000$ episodes to determine the averaged payoffs to a strategy profile.

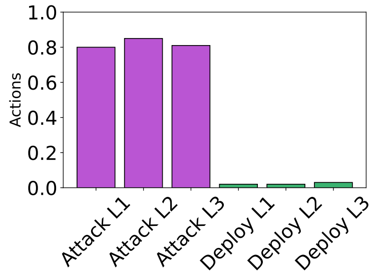
6.2 Best-Response Behaviors

We begin by examining the best-response behaviors of players against fixed opponent strategies. This serves two purposes: first, to assess whether players can produce effective best responses given diverse opponent strategies within our mixnet models, which is a critical step in PSRO, and second, to demonstrate how to interpret player response behavior within these models. To achieve these purposes, we consider four distinct types of opponent strategies: (1) random strategies, (2) strategies that focus exclusively on deploying new mixes, (3) strategies that only perform exclusion (or attacks), and (4) strategies that both exclude (or attack) and deploy new mixes in all available positions. Against each of these opponent types, we employ Stable Baselines 3 [Raffin *et al.*, 2019] to train a pure-strategy best response using ***Proximal Policy Optimization*** (PPO) [Schulman *et al.*, 2017] for both the heterogeneous model with discrete actions and the homogeneous model with continuous actions. We then interpret the resulting best-response behaviors, placing more emphasis on the homogeneous model, whose lower-dimensional action space enables more intuitive visualization and interpretation of the results.

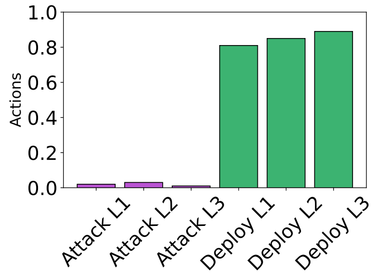


(a) Heterogeneous Model. (b) Homogeneous Model.

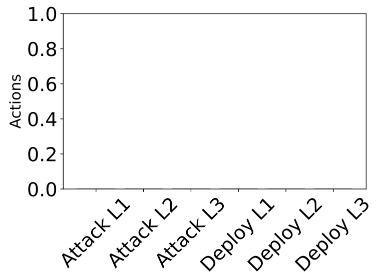
Fig. 5: Players' utilities by best responding to random strategies. Blue dots: both playing random strategies. Orange and green dots: best response vs random strategy.



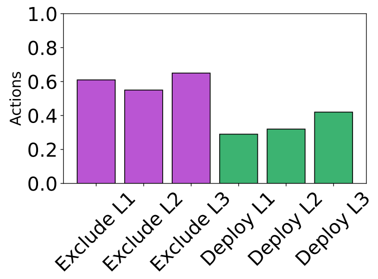
(a) BR to deploy-only strategy.



(b) BR to exclude-only strategy.



(c) BR to fully defense strategy.



(d) BR to attack-only strategy.

Fig. 6: Best responses to baseline strategies. We consider a three-layer mixnet with homogeneous mixes, represented by L1, L2, and L3. The attacker can attack and deploy on each layer while the defender can exclude and deploy accordingly.

We trained PPO strategies for both mixnet models against random opponents and found that players achieved significantly better payoffs by playing best responses instead of random strategies (Figure 5). The blue dot shows that random strategies result in poor defender performance and moderate attacker utility, highlighting the ineffectiveness of random defense. When each player responds optimally to a random opponent, the orange and green dots show improved defender utility and even higher attacker gains, respectively—demonstrating a growing utility gap and reduced system security. The attacker's utility strongly depends on the number of compromised paths. We also trained PPO in scenarios where both players take all possible actions, modeling fully committed play, and observed effective best-response behavior in both models.

In Figure 6, we show the best-response behavior in terms of the average action[9] taken on each layer within the mixnet. Specifically, Figure 6a illustrates the average action of the attacker's best response when the defender exclusively deploys new mixes without removing any existing ones. In this scenario, the attacker focuses primarily

[9] Each atomic action within a combinatorial action includes the proportion of nodes to attack/deploy on each layer. We take an average of these proportions across multiple time steps in multiple episodes.

on attacking, as any new mix deployments by the attacker are counteracted by the defender's deployments. We observed a slight positive action for mix deployment by the attacker, which can be attributed to the convergence to a suboptimal best response. Figure 6b shows the average action of the attacker's best response when the defender only removes existing mixes from the mixnet without introducing new ones. In this case, the attacker prioritizes deploying new mixes rather than attacking, as compromised mixes are continuously removed from the mixnet by the defender. In Figure 6c, the defender simultaneously removes existing mixes and deploys new ones in all available positions. This strategy neutralizes the attacker's efforts, causing the attacker to refrain from launching any further attacks (i.e., zero actions in Figure 6c). Finally, Figure 6d illustrates the average action of the defender's best response when the attacker focuses solely on attacking existing mixes. In this situation, the defender both removes compromised mixes and introduces new ones, which collectively reduce the proportion of compromised mixes in the network. All these observations indicate that developing a counter-strategy using DRL is effective against a fixed attacker on mixnets, highlighting its potential for real-world mixnet defense.

6.3 Tradeoffs among Criteria of Interest

Based on the setup in Section 6.1, we use the homogeneous model to illustrate the cost-performance tradeoffs a mechanism designer would face. These tradeoffs emerge between defense costs and the mixnet performance, measured by the number of compromised paths and bandwidth. Rather than assuming a specific set of design goals, our experiments focus on demonstrating an analysis procedure that reveals these tradeoffs. This approach allows a mechanism designer to define its own objective function and optimize it based on the tradeoffs.

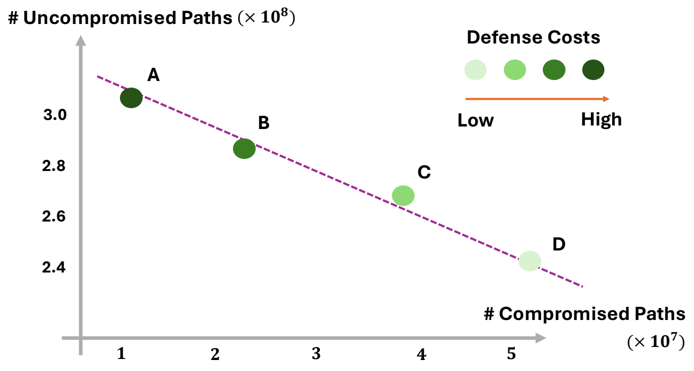


Fig. 7: Defense cost variations with respect to the number of compromised and uncompromised paths.

In Figure 7, we illustrate how defense costs, the number of compromised paths, and the number of uncompromised paths vary across different mixnet configurations, depicted as green dots. Dark green dots correspond to configurations with higher defense costs, while light green dots indicate lower defense costs. From the figure, we observe that as the security level decreases (from A to D), the number of compromised paths increases. This trend is expected, as lower security levels make mixes more vulnerable to compromise. Interestingly and somewhat counterintuitively, we also observe that defense costs tend to decrease as the security level drops. To better understand this phenomenon, we examine the breakdown of the defense costs into three components for maintenance, exclusion, and deployment. These are shown in Figure 8, Figure 9, and Figure 10, respectively.

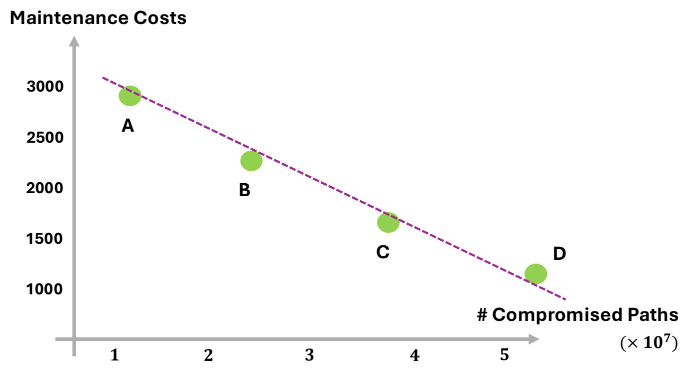


Fig. 8: Maintenance cost variations with respect to the number of compromised paths.

Figure 8 illustrates how maintenance costs vary with different configurations and the resulting number of compromised paths. We observe that as the security level decreases, maintenance costs also decline. This is primarily because lower-security mixes incur significantly lower maintenance costs, leading to a reduction in overall expenses. Furthermore, as we show later, this trend is also driven by reduced deployment, that is, fewer mixes are deployed at lower security levels, further decreasing maintenance costs.

Figure 9 shows how exclusion costs change under different configurations and the corresponding number of compromised paths. We find that as the security level decreases, exclusion costs increase. This trend is intuitive: lower security levels lead to a higher proportion of compromised paths, requiring the defender to exert more effort by excluding a greater number of mixes from the mixnet and ultimately resulting in higher exclusion costs. Moreover, we observe that exclusion costs rise most sharply when the security level drops from the highest setting (A to B), with the rate of increase diminishing as the security level continues to decline (from B to D). This suggests that defensive efforts must be significantly increased when the highest level of security is relaxed.

Figure 10 shows how deployment costs vary across different configurations. We observe that as the security level decreases, deployment costs also decline. The drop in deployment costs occurs because the defender strategically limits new deployments,

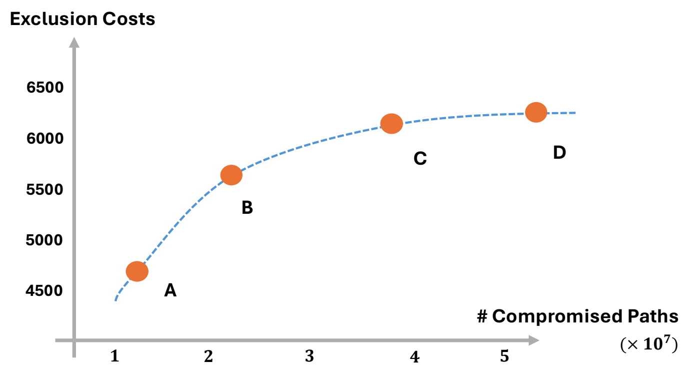


Fig. 9: Exclusion cost variations with respect to the number of compromised paths.

knowing they are likely to be quickly compromised, which would increase both penalties and maintenance expenses. Nonetheless, the defender must still deploy a sufficient number of mixes to ensure the mixnet remains functional. This reduction in deployment leads to lower maintenance costs, bandwidth, and eventually, lower total defense costs.

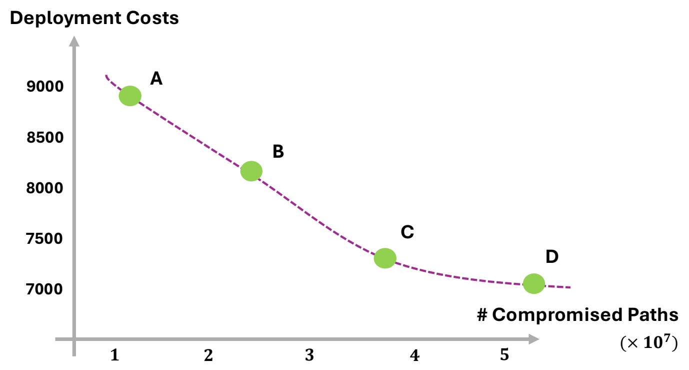


Fig. 10: Deployment cost variations with respect to the number of compromised paths.

Figure 11 illustrates the relationship between bandwidth and the number of compromised paths under various configurations. We observe that bandwidth decreases as the severity of the attack increases. This indicates that a powerful attacker can undermine the mixnet in two significant ways. First, by compromising more paths, the attacker weakens anonymity since more user messages become traceable, increasing the risk of identity exposure. This effect is also evident in the reduced number of uncompromised paths shown in Figure 7. Second, a strong attacker can degrade the mixnet's overall bandwidth, potentially introducing substantial delays for all users.

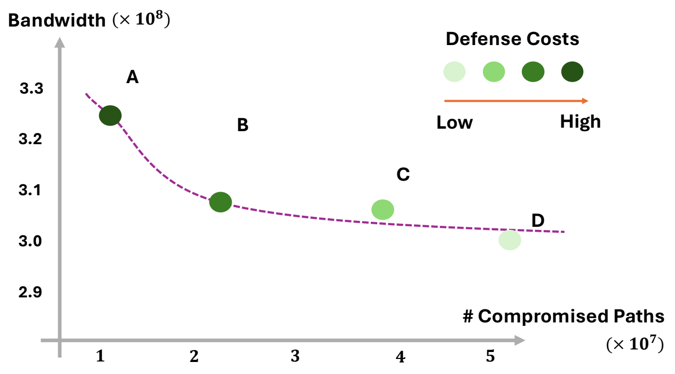


Fig. 11: Bandwidth variations with the number of compromised paths.

Our analysis highlights the tradeoffs and key insights that arise in the mechanism design of a mixnet. By understanding these interdependencies, a mechanism designer can anticipate the defender's strategic behavior, evaluate the cost-performance of various configurations, and identify optimal design choices that balance security, cost, and functionality in adversarial environments.

7 Conclusion and Discussion

In this study, we propose an empirical mechanism design framework for designing mixnets using PSRO. To model different structural complexities, we introduce two variants of the mixnet game: the heterogeneous and homogeneous mixnet games, where a centralized attacker and defender vie for control over the mixnet. Through empirical analysis, we reveal the underlying cost-performance tradeoffs, illustrating how these tradeoffs can inform a mechanism designer seeking to balance deployment cost against system performance.

A promising research direction is to create a multiscale mixnet model that spans the spectrum between fully heterogeneous and fully homogeneous representations. In realistic settings, only a few critical mixes warrant fine-grained, game-theoretic treatment, while the rest can be modeled more coarsely. For mechanism design, pursuing this approach could help pinpoint which mixes should receive heterogeneous modeling to best preserve the mixnet's overall security.

Bibliography

Enrique Areyan Viqueira, Amy Greenwald, Cyrus Cousins, and Eli Upfal. Learning simulation-based games from data. In *AAMAS*, 2019.

Iness Ben Guirat, Devashish Gosain, and Claudia Diaz. Mixim: Mixnet design decisions and empirical evaluation. In *Workshop on Privacy in the Electronic Society*, 2021.

Lucas Benedicic, Martin Benedikt Busch, Sebastian Bürgel, Robert Kiel, and Rich McDowell. HOPR as the transport layer of privacy-aware, GDPR-compliant IoT health systems, 2022.

Ariyan Bighashdel, Yongzhao Wang, Stephen McAleer, Rahul Savani, and Frans A Oliehoek. Policy space response oracles: A survey. In *IJCAI*, 2024.

Erik Brinkman and Michael P Wellman. Empirical mechanism design for optimizing clearing interval in frequent call markets. In *EC*, 2017.

David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 1981.

Henry Corrigan-Gibbs, David Isaac Wolinsky, and Bryan Ford. Proactively accountable anonymous messaging in verdict. In *USENIX Security*, 2013.

Claudia Diaz, Harry Halpin, and Aggelos Kiayias. The nym network: The next generation of privacy infrastructure. 2021.

Tariq Elahi, John A Doucette, Hadi Hosseini, Steven J Murdoch, and Ian Goldberg. A framework for the game-theoretic analysis of censorship resistance. *Privacy Enhancing Technologies*, 2016.

Madelyn Gatchel and Bryce Wiedenbeck. Learning parameterized families of games. In *AAMAS*, 2023.

Iness Ben Guirat and Claudia Diaz. Mixnet optimization methods. *Privacy Enhancing Technologies*, 2022.

Harry Halpin and Andrei Serjantov. Incentives and censorship resistance for mixnets revisited. In *Cambridge International Workshop on Security Protocols*, pages 64–69. Springer, 2023.

HOPR. The Book of HOPR. https://hoprnet.org/Book_Of_Hopr_2021.01_v1.pdf, 2021.

Patrick R. Jordan, Michael P. Wellman, and Guha Balakrishnan. Strategy and mechanism lessons from the first ad auctions trading agent competition. In *EC*, 2010.

Marc Lanctot, Vinicius Zambaldi, Audrūnas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *Neurips*, 2017.

David Lazar and Nickolai Zeldovich. Alpenhorn: Bootstrapping secure communication without leaking metadata. In *OSDI*, 2016.

Zun Li, Marc Lanctot, Kevin R McKee, Luke Marris, Ian Gemp, Daniel Hennes, Paul Muller, Kate Larson, Yoram Bachrach, and Michael P Wellman. Combining tree-search, generative models, and nash bargaining concepts in game-theoretic reinforcement learning. *arXiv preprint arXiv:2302.00797*, 2023.

Xinshu Ma, Florentin Rochet, and Tariq Elahi. Stopping silent sneaks: Defending against malicious mixes with topological engineering. In *ACSAC*, 2022.

Ania M Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. The loopix anonymity system. In *USENIX Security Symposium*, 2017.

Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. Stable baselines3. https://github.com/DLR-RM/stable-baselines3, 2019.

Mahdi Rahimi, Piyush Kumar Sharma, and Claudia Diaz. Larmix: Latency-aware routing in mix networks. In *The Network and Distributed System Security Symposium. Internet Society*, 2024.

Rahul Savani and Theodore L Turocy. Gambit: The package for computation in game theory. 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Mohammad Hassan Mojtahed Soleimani, Muharram Mansoorizadeh, and Mohammad Nassiri. Real-time identification of three tor pluggable transports using machine learning techniques. *The Journal of Supercomputing*, 2018.

Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. StarCraft II: A new challenge for reinforcement learning. Technical Report 1708.04782v1, arXiv, 2017.

Yevgeniy Vorobeychik, Christopher Kiekintveld, and Michael P Wellman. Empirical mechanism design: Methods, with application to a supply-chain scenario. In *Proceedings of the 7th ACM conference on Electronic commerce*, pages 306–315, 2006.

Yevgeniy Vorobeychik, Daniel M. Reeves, and Michael P. Wellman. Constrained automated mechanism design for infinite games of incomplete information. *Autonomous Agents and Multi-Agent Systems*, 25:313–351, 2012.

Yongzhao Wang and Michael P Wellman. Regularization for strategy exploration in empirical game-theoretic analysis. *arXiv preprint arXiv:2302.04928*, 2023.

Michael P. Wellman. Methods for empirical game-theoretic analysis (extended abstract). In *AAAI*, 2006.

David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. Dissent in numbers: Making strong anonymity scale. In *10th USENIX Symposium on Operating Systems Design and Implementation*, 2012.

David Isaac Wolinsky, Henry Corrigan-Gibbs, Bryan Ford, and Aaron Johnson. Scalable anonymous group communication in the anytrust model. In *EuroSec*, 2012.

xx Network. White Paper xx cMix. https://xx.network/wp-content/uploads/2021/10/xxcMixwhitepaper.pdf, 2021.