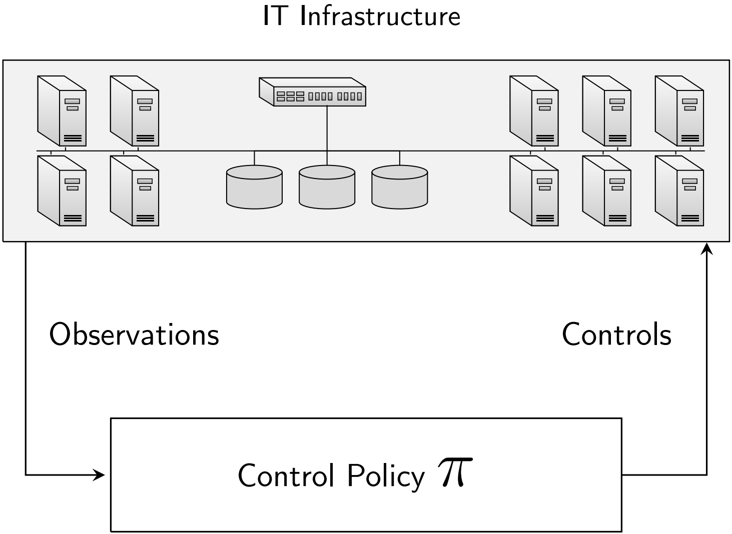
Self-Learning Systems for Cyber Security

NSE Seminar

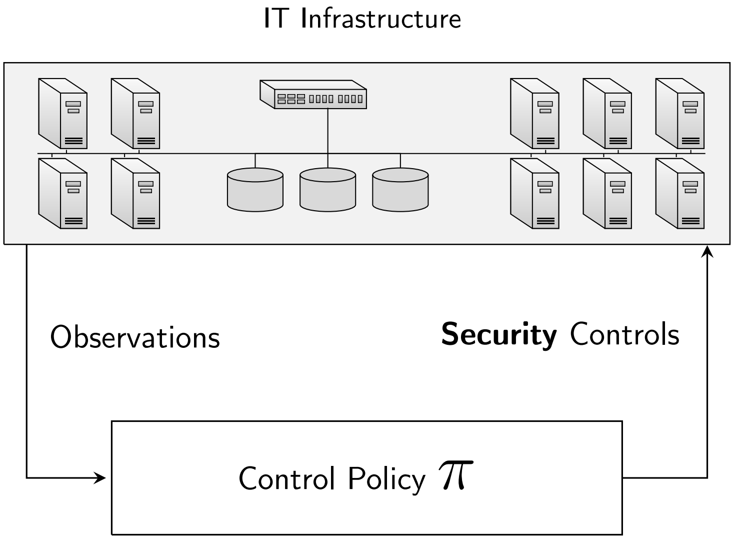
Kim Hammar & Rolf Stadler

December 4, 2020

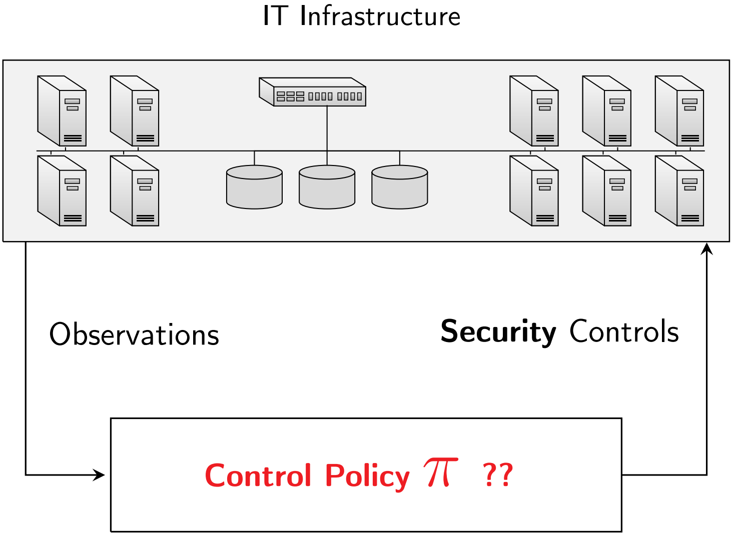
Self-Learning Systems for Cyber Security



Self-Learning Systems for Cyber Security



Self-Learning Systems for Cyber Security



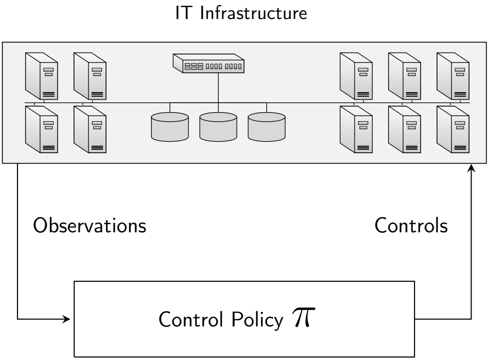
IT Infrastructure

Observations

Security Controls

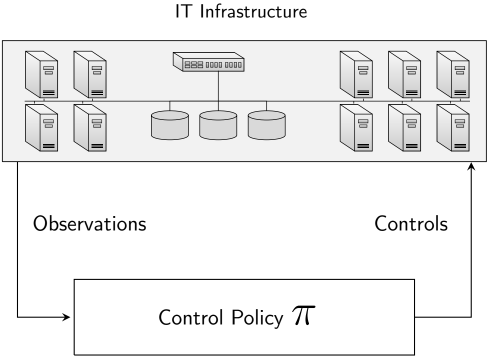
Control Policy π **??**

Self-Learning Systems for Cyber Security



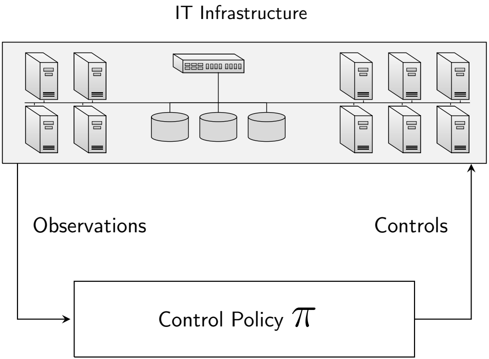
- **What are useful controls**?
  - Penetration tests
  - Intrusion prevention strategies & Adaptive security policies
  - Limiting virus spread
  - ⋮

Self-Learning Systems for Cyber Security



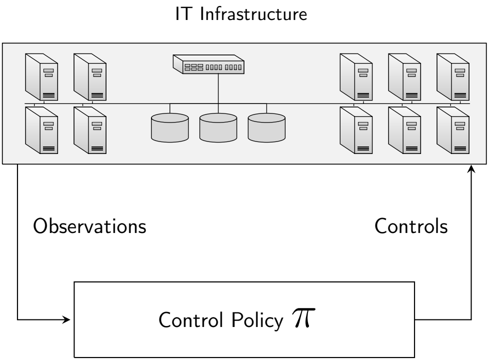
- **What are useful controls**?
  - Penetration tests
  - Intrusion prevention strategies & Adaptive security policies
  - Limiting virus spread
  - ⋮

Self-Learning Systems for Cyber Security



- **What are useful controls**?
  - Penetration tests
  - Intrusion prevention strategies & Adaptive security policies
  - Limiting virus spread
  - ⋮

Self-Learning Systems for Cyber Security



- **What are useful controls**?
  - Penetration tests
  - Intrusion prevention strategies & Adaptive security policies
  - Limiting virus spread
  - ⋮

Related Work

- **Model-Based Approaches**:
  - Optimal Control[1]
  - Dynamic Programming[2]
  - Computational Game Theory[3]
- **Simulation-Based Approaches**
  - Evolutionary Methods[4]
  - Reinforcement Learning[5]

[1] Jianguo Ren, Yonghong Xu, and Chunming Zhang. "Optimal Control of a Delay-Varying Computer Virus Propagation Model". In: *Discrete Dynamics in Nature and Society* 2013 (2013), p. 210291. ISSN: 1026-0226. DOI: 10.1155/2013/210291. URL: https://doi.org/10.1155/2013/210291.

[2] Mohammad Rasouli, Erik Miehling, and Demosthenis Teneketzis. "A Supervisory Control Approach to Dynamic Cyber-Security". In: *Decision and Game Theory for Security*. Ed. by Radha Poovendran and Walid Saad. Cham: Springer International Publishing, 2014, pp. 99–117. ISBN: 978-3-319-12601-2.

[3] Marten van Dijk et al. "FlipIt: The Game of "Stealthy Takeover"". In: *Journal of Cryptology* 26.4 (2013), pp. 655–713. ISSN: 1432-1378. DOI: 10.1007/s00145-012-9134-5. URL: https://doi.org/10.1007/s00145-012-9134-5, Tansu Alpcan and Tamer Basar. *Network Security: A Decision and Game-Theoretic Approach*. 1st. USA: Cambridge University Press, 2010. ISBN: 0521119324.

[4] R. Bronfman-Nadas, N. Zincir-Heywood, and J. T. Jacobs. "An Artificial Arms Race: Could it Improve Mobile Malware Detectors?" In: *2018 Network Traffic Measurement and Analysis Conference (TMA)*. 2018, pp. 1–8.

[5] Richard Elderman et al. "Adversarial Reinforcement Learning in a Cyber Security Simulation". In: *ICAART*. 2017, Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Related Work

- **Model-Based Approaches**:
  - Optimal Control[6]
  - Dynamic Programming[7]
  - Computational Game Theory[8]
- Simulation-Based Approaches
  - Evolutionary Methods[9]
  - Reinforcement Learning[10]

[6] Jianguo Ren, Yonghong Xu, and Chunming Zhang. "Optimal Control of a Delay-Varying Computer Virus Propagation Model". In: *Discrete Dynamics in Nature and Society* 2013 (2013), p. 210291. ISSN: 1026-0226. DOI: 10.1155/2013/210291. URL: https://doi.org/10.1155/2013/210291.

[7] Mohammad Rasouli, Erik Miehling, and Demosthenis Teneketzis. "A Supervisory Control Approach to Dynamic Cyber-Security". In: *Decision and Game Theory for Security*. Ed. by Radha Poovendran and Walid Saad. Cham: Springer International Publishing, 2014, pp. 99–117. ISBN: 978-3-319-12601-2.

[8] Marten van Dijk et al. "FlipIt: The Game of "Stealthy Takeover"". In: *Journal of Cryptology* 26.4 (2013), pp. 655–713. ISSN: 1432-1378. DOI: 10.1007/s00145-012-9134-5. URL: https://doi.org/10.1007/s00145-012-9134-5, Tansu Alpcan and Tamer Basar. *Network Security: A Decision and Game-Theoretic Approach*. 1st. USA: Cambridge University Press, 2010. ISBN: 0521119324.

[9] R. Bronfman-Nadas, N. Zincir-Heywood, and J. T. Jacobs. "An Artificial Arms Race: Could it Improve Mobile Malware Detectors?" In: *2018 Network Traffic Measurement and Analysis Conference (TMA)*. 2018, pp. 1–8.

[10] Richard Elderman et al. "Adversarial Reinforcement Learning in a Cyber Security Simulation". In: *ICAART*. 2017, Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Related Work

- Model-Based Approaches:
  - Optimal Control[11]
  - Dynamic Programming[12]
  - Computational Game Theory[13]
- **Simulation-Based Approaches**
  - Evolutionary Methods[14]
  - Reinforcement Learning[15]

[11] Jianguo Ren, Yonghong Xu, and Chunming Zhang. "Optimal Control of a Delay-Varying Computer Virus Propagation Model". In: *Discrete Dynamics in Nature and Society* 2013 (2013), p. 210291. ISSN: 1026-0226. DOI: 10.1155/2013/210291. URL: https://doi.org/10.1155/2013/210291.

[12] Mohammad Rasouli, Erik Miehling, and Demosthenis Teneketzis. "A Supervisory Control Approach to Dynamic Cyber-Security". In: *Decision and Game Theory for Security*. Ed. by Radha Poovendran and Walid Saad. Cham: Springer International Publishing, 2014, pp. 99–117. ISBN: 978-3-319-12601-2.

[13] Marten van Dijk et al. "FlipIt: The Game of "Stealthy Takeover"". In: *Journal of Cryptology* 26.4 (2013), pp. 655–713. ISSN: 1432-1378. DOI: 10.1007/s00145-012-9134-5. URL: https://doi.org/10.1007/s00145-012-9134-5, Tansu Alpcan and Tamer Basar. *Network Security: A Decision and Game-Theoretic Approach*. 1st. USA: Cambridge University Press, 2010. ISBN: 0521119324.

[14] R. Bronfman-Nadas, N. Zincir-Heywood, and J. T. Jacobs. "An Artificial Arms Race: Could it Improve Mobile Malware Detectors?" In: *2018 Network Traffic Measurement and Analysis Conference (TMA)*. 2018, pp. 1–8.

[15] Richard Elderman et al. "Adversarial Reinforcement Learning in a Cyber Security Simulation". In: *ICAART*. 2017, Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Related Work

- Model-Based Approaches:
  - Optimal Control[16]
  - Dynamic Programming[17]
  - Computational Game Theory[18]
- **Simulation-Based Approaches**
  - Evolutionary Methods[19]
  - Reinforcement Learning[20]

[16] Jianguo Ren, Yonghong Xu, and Chunming Zhang. "Optimal Control of a Delay-Varying Computer Virus Propagation Model". In: *Discrete Dynamics in Nature and Society* 2013 (2013), p. 210291. ISSN: 1026-0226. DOI: 10.1155/2013/210291. URL: https://doi.org/10.1155/2013/210291.

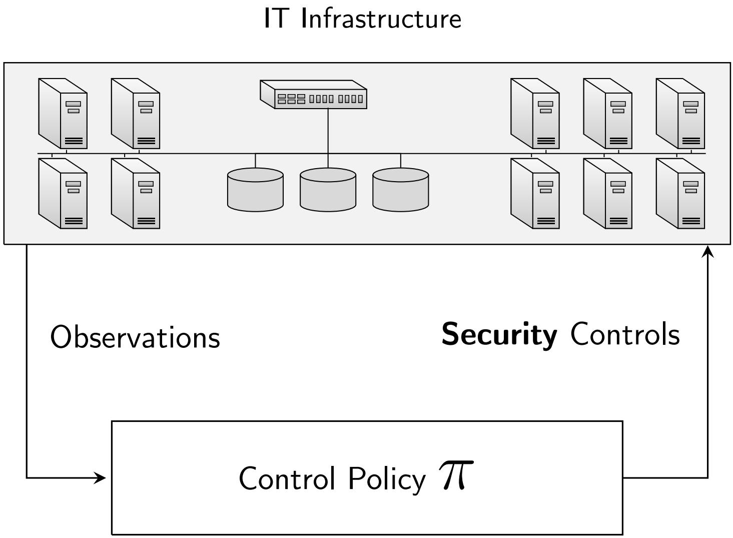
[17] Mohammad Rasouli, Erik Miehling, and Demosthenis Teneketzis. "A Supervisory Control Approach to Dynamic Cyber-Security". In: *Decision and Game Theory for Security*. Ed. by Radha Poovendran and Walid Saad. Cham: Springer International Publishing, 2014, pp. 99–117. ISBN: 978-3-319-12601-2.

[18] Marten van Dijk et al. "FlipIt: The Game of "Stealthy Takeover"". In: *Journal of Cryptology* 26.4 (2013), pp. 655–713. ISSN: 1432-1378. DOI: 10.1007/s00145-012-9134-5. URL: https://doi.org/10.1007/s00145-012-9134-5, Tansu Alpcan and Tamer Basar. *Network Security: A Decision and Game-Theoretic Approach*. 1st. USA: Cambridge University Press, 2010. ISBN: 0521119324.

[19] R. Bronfman-Nadas, N. Zincir-Heywood, and J. T. Jacobs. "An Artificial Arms Race: Could it Improve Mobile Malware Detectors?" In: *2018 Network Traffic Measurement and Analysis Conference (TMA)*. 2018, pp. 1–8.

[20] Richard Elderman et al. "Adversarial Reinforcement Learning in a Cyber Security Simulation". In: *ICAART*. 2017, Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Model-Based Control: DT Dynamical System Model



IT Infrastructure

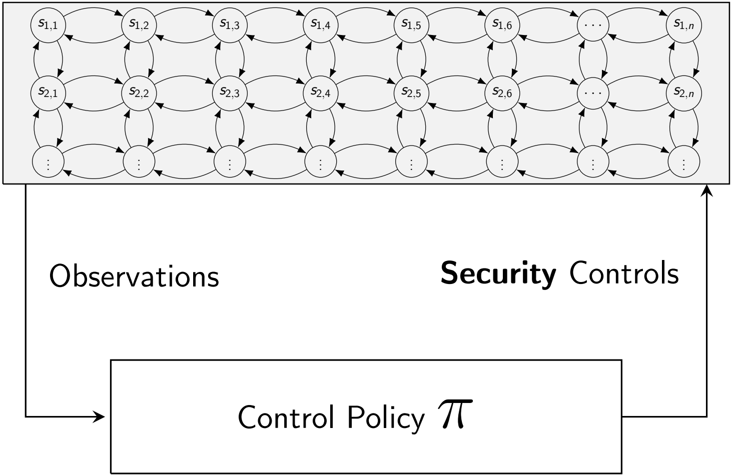
Observations

Security Controls

Control Policy π

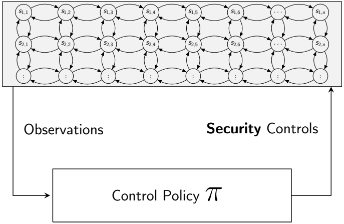
Model-Based Control: DT Dynamical System Model

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^{a}_{ss'}, \mathcal{R}^{a}_{ss'}, \gamma, \rho_0, T \rangle$$



Model-Based Control: DT Dynamical System Model

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma, \rho_0, T \rangle$$


$s_{1,1}$ $s_{1,2}$ $s_{1,3}$ $s_{1,4}$ $s_{1,5}$ $s_{1,6}$ $\cdots$ $s_{1,n}$
$s_{2,1}$ $s_{2,2}$ $s_{2,3}$ $s_{2,4}$ $s_{2,5}$ $s_{2,6}$ $\cdots$ $s_{2,n}$
$\vdots$

Observations

Security Controls

Control Policy π

- $\mathcal{M}$ = Markov Decision Process
- **Problem reduces to solving Bellman's equations**

$$u_t(h_t) = \sup_{a \in A_{s_t}} \left[r_t(s_t, a) + \sum_{j \in S} p_t(j|s_t, a) \underbrace{u_{t+1}(h_t, a, j)}_{\text{-cost to go}} \right]$$

- Solution methods[21]: Backward induction, Dynamic programming (Value iteration, Policy iteration)

[21] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. 1st. USA: John Wiley and Sons, Inc., 1994. ISBN: 0471619779.

Limitations of the Model-Based Approach

Modeling Challenge

How to model complex systems and cyber attacks **accurately**?

Scalability Challenge

Models are often impractical due to scale of applications.

- e.g. assume MDP model of cyber range: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma, \rho_0, T \rangle$
- Need to solve:

$$V^*(s) = \max_a \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^*(s') \right]$$

Limitations of the Model-Based Approach

Modeling Challenge

How to model complex systems and cyber attacks **accurately**?

Scalability Challenge

Models are often impractical due to scale of applications.

- e.g. assume MDP model of cyber range: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma, \rho_0, T \rangle$
- Need to solve:

$$V^*(s) = \max_a \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^*(s') \right]$$

Limitations of the Model-Based Approach

Modeling Challenge

How to model complex systems and cyber attacks **accurately**?

Scalability Challenge

Models are often impractical due to scale of applications.

- e.g. assume MDP model of cyber range: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma, \rho_0, T \rangle$
- Need to solve:

$$V^*(s) = \max_a \sum_{s' \in \mathcal{S}} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^*(s') \right]$$

Limitations of the Model-Based Approach

Modeling Challenge

How to model complex systems and cyber attacks **accurately**?

Scalability Challenge

Models are often impractical due to scale of applications.

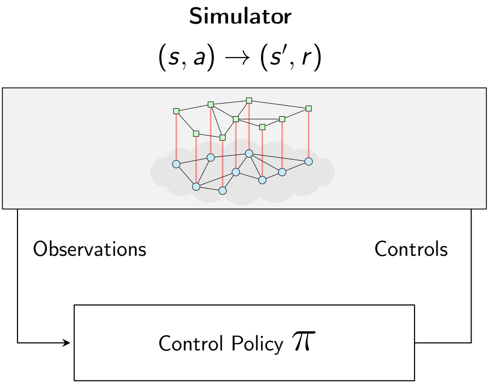
- e.g. assume MDP model of cyber range: $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma, \rho_0, T \rangle$
- Need to solve (**curse of modeling**[22]):

$$V^*(s) = \max_a \sum_{s' \in \boxed{\mathcal{S}}} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^*(s')\right]$$

$|\mathcal{S}| = 10^{170}$ (Atoms in the universe $\approx 10^{80}$)

[22] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. 1st. Athena Scientific, 1996. ISBN: 1886529108.

Simulation-Based Approaches



Simulator

$(s, a) \to (s', r)$

Observations

Controls

Control Policy π

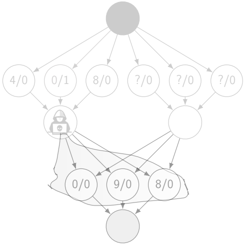
- Rather than defining complete model $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}^a_{ss'}, \mathcal{R}^a_{ss'}, \gamma, \rho_0, T \rangle \implies$ define simulator that can be sampled from.
- **Pros:** scalable, simple to implement, flexible
- **Cons:** (same as model-based) is it realistic??

Simulation-Based Example: Intrusion Prevention[23]

Question

Can effective security-strategies emerge from self-play RL?

- Model network as graph $\mathcal{G} = \langle \mathcal{N}, \mathcal{E} \rangle$
- Attack/defense attributes per node $S_k = \langle S_k^A, S_k^D \rangle$
- Simulate outcome of actions as function $f(s, a)$.
- Partially observed two-player Markov game



- Results:
  - Challenging learning task but possible
  - ϵ-optimal strategies emerge using our proposed method
    - AR policy, opponent pool, PPO, function approximation
  - Strategies are abstract, cannot easily be verified

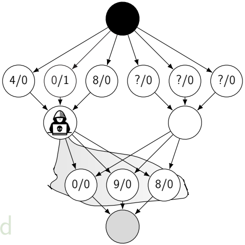
[23] Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Simulation-Based Example: Intrusion Prevention[24]

Question

Can effective security-strategies emerge from self-play RL?

- Model network as graph $\mathcal{G} = \langle \mathcal{N}, \mathcal{E} \rangle$
- Attack/defense attributes per node $S_k = \langle S_k^A, S_k^D \rangle$
- Simulate outcome of actions as function $f(s, a)$.
- Partially observed two-player Markov game



- Results:
    - Challenging learning task but possible
    - ϵ-optimal strategies emerge using our proposed method
        - AR policy, opponent pool, PPO, function approximation
    - Strategies are abstract, cannot easily be verified

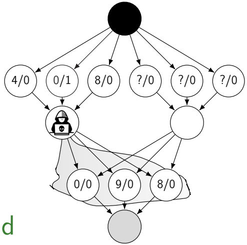
[24] Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Simulation-Based Example: Intrusion Prevention[25]

Question

Can effective security-strategies emerge from self-play RL?

- Model network as graph $\mathcal{G} = \langle \mathcal{N}, \mathcal{E} \rangle$
- Attack/defense attributes per node $S_k = \langle S_k^A, S_k^D \rangle$
- Simulate outcome of actions as function $f(s, a)$.
- Partially observed two-player Markov game

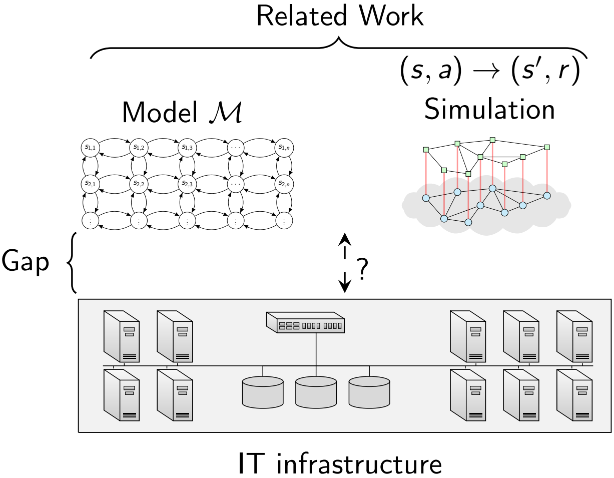


- **Results**:
  - Challenging learning task but possible
  - ϵ-optimal strategies emerge using our proposed method
    - AR policy, opponent pool, PPO, function approximation
  - Strategies are abstract, cannot easily be verified

[25] Kim Hammar and Rolf Stadler. "Finding Effective Security Strategies through Reinforcement Learning and Self-Play". In: *International Conference on Network and Service Management (CNSM 2020) (CNSM 2020)*. Izmir, Turkey, Nov. 2020.

Research Questions

- Prior work focused on **simulation-based** and **model-based** approaches
  - *Assumed to be impractical to interact with real systems*



Related Work

$(s, a) \rightarrow (s', r)$

Model $\mathcal{M}$

Simulation

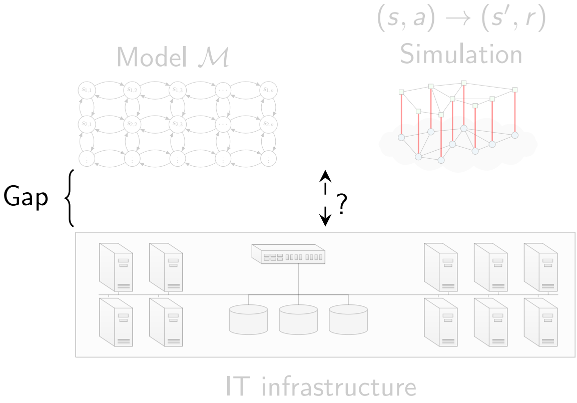
Gap

?

IT infrastructure

Research Questions

- How large is this gap? How can we bridge it?
- Take inspiration from early works studying this problem[26]



$(s, a) \rightarrow (s', r)$

Simulation

Model $\mathcal{M}$

Gap

?

IT infrastructure

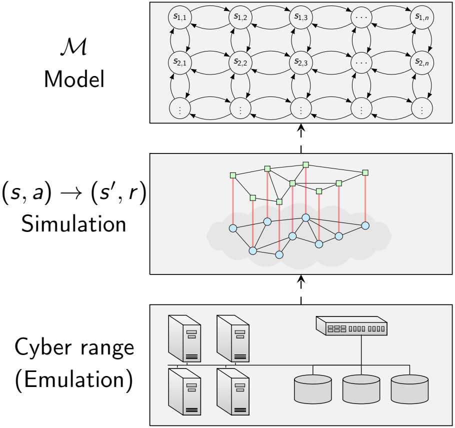
[26] Gabriel Dulac-Arnold, Daniel J. Mankowitz, and Todd Hester. "Challenges of Real-World Reinforcement Learning". In: *CoRR* abs/1904.12901 (2019). arXiv: 1904.12901. URL: http://arxiv.org/abs/1904.12901, Hyrum S. Anderson et al. "Learning to Evade Static PE Machine Learning Malware Models via Reinforcement Learning". In: *CoRR* abs/1801.08917 (2018). arXiv: 1801.08917. URL: http://arxiv.org/abs/1801.08917, Piotr Gawlowicz and Anatolij Zubow. "ns3-gym: Extending OpenAI Gym for Networking Research". In: *CoRR* abs/1810.03943 (2018). arXiv: 1810.03943. URL: http://arxiv.org/abs/1810.03943.

Research Questions

Assumption

"Assumed to be impractical to interact with real systems"

- **Can we question this assumption?**
  - What is the right balance between model/simulation/real system?



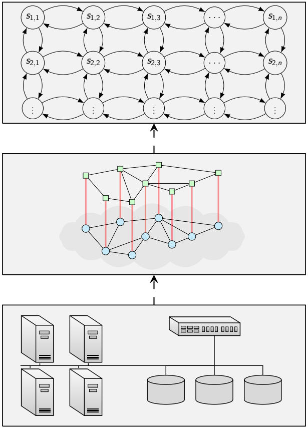
$\mathcal{M}$
Model

$(s, a) \rightarrow (s', r)$
Simulation

Cyber range
(Emulation)

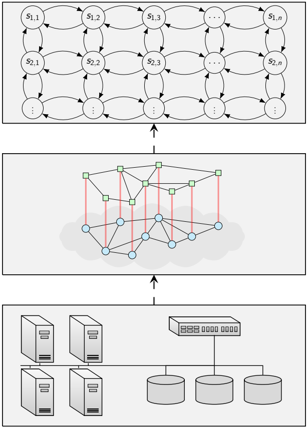
Our Approach

- **Goals**:
  - Framework for learning control tasks in security
  - Connect simulations & models with practical environment
- What is a good environment for evaluation?
  - Cyber ranges
  - Used to evaluate human security experts



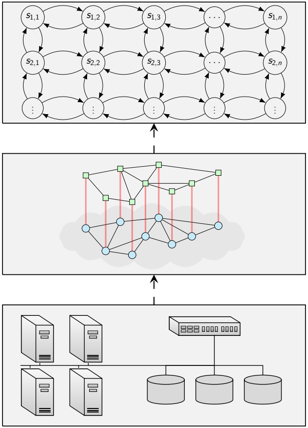
Our Approach

- Goals:
  - Framework for learning control tasks in security
  - Connect simulations & models with practical environment
- **What is a good environment for evaluation?**
  - Cyber ranges
  - Used to evaluate human security experts



Our Approach

- **Goals**:
  - Framework for learning control tasks in security
  - Connect simulations & models with practical environment
- **What is a good environment for evaluation?**
  - Cyber ranges
  - Used to evaluate human security experts

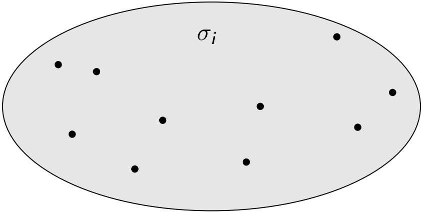


Idea

Need a **tool to generate cyber** ranges for different control tasks

Generation of Cyber Ranges for Control Tasks

$\Sigma = \langle \mathcal{C}, \mathcal{O}, \mathcal{S}, \mathcal{U}, \mathcal{T} \rangle$ Configuration Space

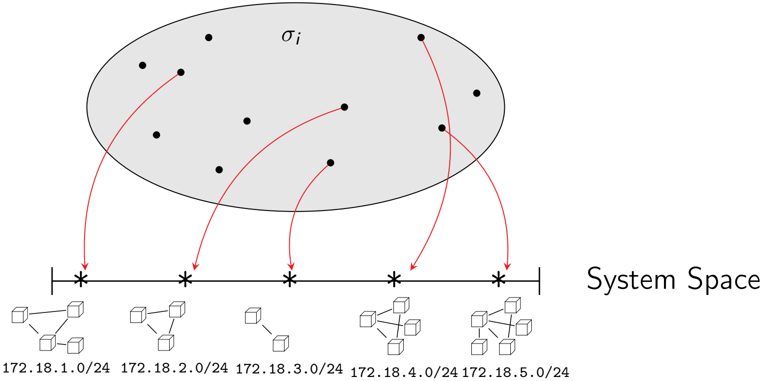


- The **configuration space**[27] defines the networks that can be generated.
  - Controls $\mathcal{C}$ (e.g. nmap, firewall configs, metasploit, etc.)
  - Operating Systems $\mathcal{O}$ (e.g. Kali, Ubuntu 20, etc.)
  - Services $\mathcal{S}$ (e.g. Kafka, MongoDB, NTP, etc.)
  - User types $\mathcal{U}$ (e.g. root, non-root, various groups)
  - Topologies $\mathcal{T}$ (implemented using firewall rules)

[27] implemented with a set of **docker images**

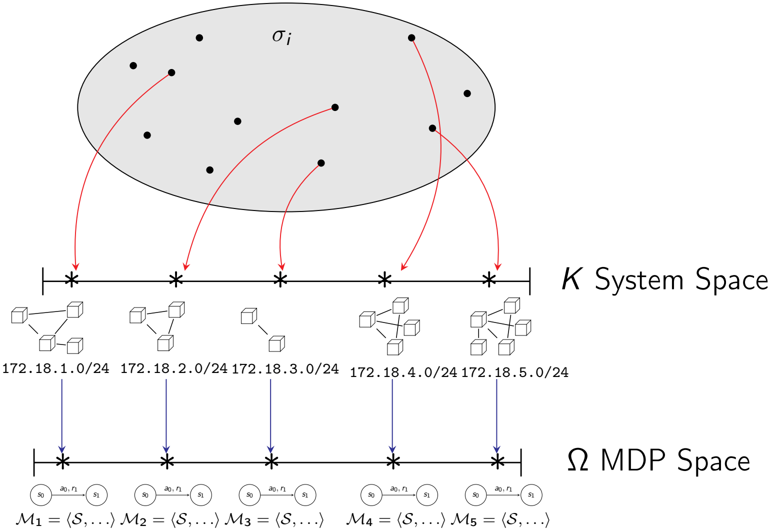
Generation of Cyber Ranges for Control Tasks

$\Sigma = \langle \mathcal{C}, \mathcal{O}, \mathcal{S}, \mathcal{U}, \mathcal{T} \rangle$ Configuration Space



Generation of Cyber Ranges for Control Tasks

$\Sigma = \langle \mathcal{C}, \mathcal{O}, \mathcal{S}, \mathcal{U}, \mathcal{T} \rangle$ Configuration Space



σ_i

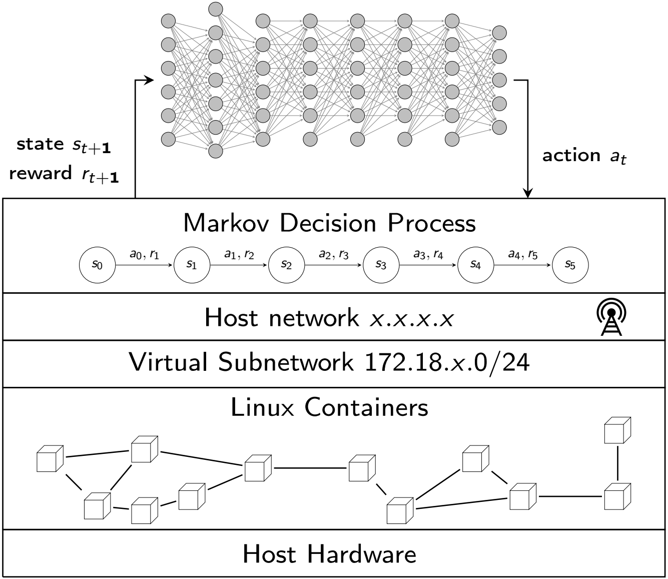
K System Space

172.18.1.0/24 172.18.2.0/24 172.18.3.0/24 172.18.4.0/24 172.18.5.0/24

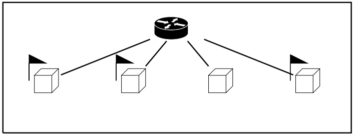
Ω MDP Space

$\mathcal{M}_1 = \langle \mathcal{S}, \ldots \rangle$ $\mathcal{M}_2 = \langle \mathcal{S}, \ldots \rangle$ $\mathcal{M}_3 = \langle \mathcal{S}, \ldots \rangle$ $\mathcal{M}_4 = \langle \mathcal{S}, \ldots \rangle$ $\mathcal{M}_5 = \langle \mathcal{S}, \ldots \rangle$

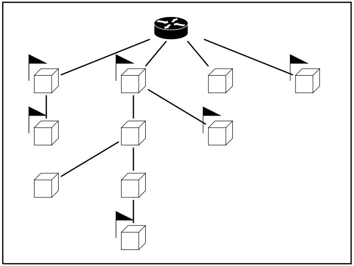
System Architecture



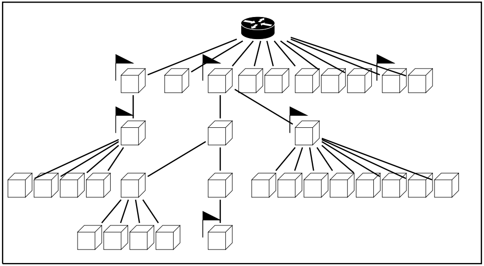
First Evaluation of Framework: **Learn** to Capture the Flag



Learning task v_1



Learning task v_2

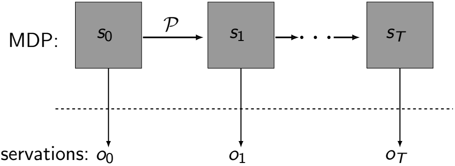


Learning task v_3

System Model (1/3)

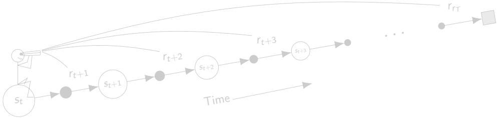
- **Hidden Markov Model**. The agent estimates the state of the system based on a sequence of observations $o_1, o_2, \ldots \in \mathcal{O}$

System Partial Observability



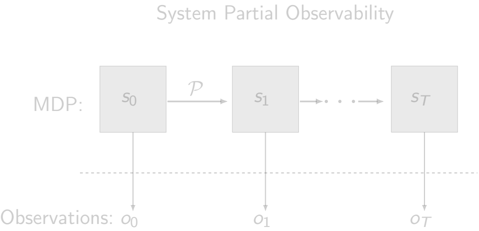
- **Infinite Discounted Time-Horizon**. Discrete time, decision epochs $T = \mathbb{N}_{\geq 0}$. Objective:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=1}^{\infty} \lambda^{t-1} r(s_t, a_t)\right]$$



System Model (1/3)

- **Hidden Markov Model**. The agent estimates the state of the system based on a sequence of observations $o_1, o_2, \ldots \in \mathcal{O}$



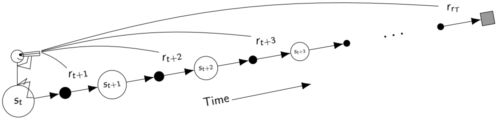
System Partial Observability

MDP: s_0 $\xrightarrow{\mathcal{P}}$ s_1 $\rightarrow \cdots \rightarrow$ s_T

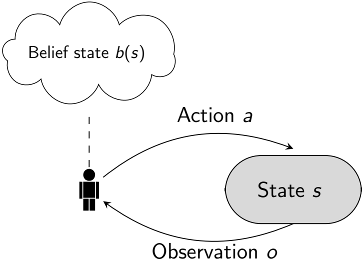
Observations: o_0 o_1 o_T

- **Infinite Discounted Time-Horizon**. Discrete time, decision epochs $T = \mathbb{N}_{\geq 0}$. Objective:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=1}^{\infty} \lambda^{t-1} r(s_t, a_t)\right]$$



System Model (2/3)



Let b_t be the belief state at time t

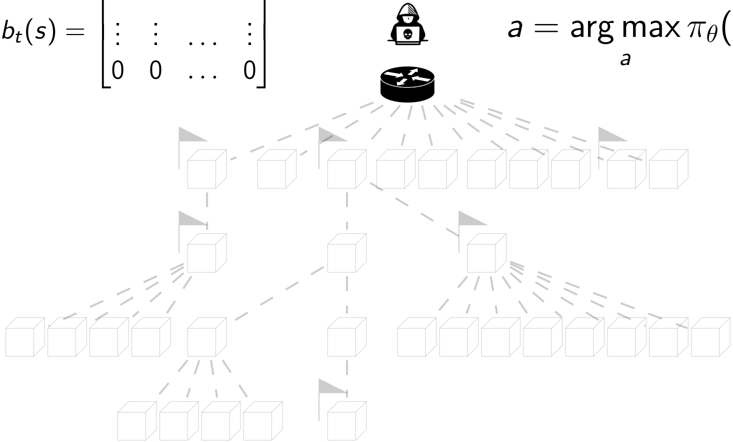
$$b_t(s) = \mathbb{P}[s_{t+1} = s | b_t], \quad s = \begin{bmatrix} p_{1,1} & p_{1,2} & \dots & sh_1 \\ \vdots & \vdots & \dots & \vdots \\ p_{N,1} & p_{N,2} & \dots & sh_N \end{bmatrix} \in \mathcal{S} \in \mathbb{R}^{N \times 34}$$

state estimated based on **basis functions** $\{\phi_1, \dots, \phi_{34}\}$ from observations o. e.g.

$$\phi_1(o) = \mathbb{1}_{\text{port 22 open}} \quad \phi_2(o) = \mathbb{1}_{\text{shell access}} \quad \dots \quad \phi_{34}(o) = \#\text{CVEs}.$$

System Model 2/3

$$b_t(s) = \begin{bmatrix} 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ldots & \vdots \\ 0 & 0 & \ldots & 0 \end{bmatrix}$$



$$a = \arg\max_{a} \pi_\theta(a|s)$$

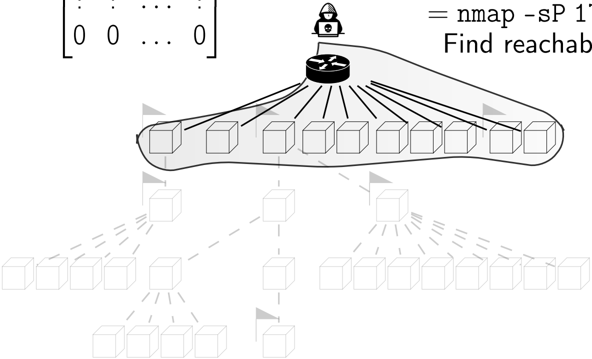
System Model 2/3

$$b_t = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$a_t = \arg\max_a \pi_\theta(a|b_t)$$

$= $ `nmap -sP 172.18.3.0/24`

Find reachable machines on network



System Model (2/3)

- Let $\mathcal{A} \triangleq \{\texttt{nmap}_i, \texttt{metasploit}_i, \texttt{nikto}_i, \ldots\}$ be the **action space**. $\mathcal{A} \subset \mathcal{B}$ where $\mathcal{B}$ is the set of commands of the Bash command-line.

- Let

$$r(s_{t+\mathbf{1}} | a_t, s_t) = \begin{cases} 10 & \text{if } b_{t+\mathbf{1}}^{\mathbf{\#flags}} > b_t^{\mathbf{\#flags}} \\ 0 & \text{if } b_{t+\mathbf{1}} \neq b_t \\ -10 & \text{otherwise} \end{cases}$$

  be the **reward function**, realizing the agent's objective to capture the flags in the system.

System Model (2/3)

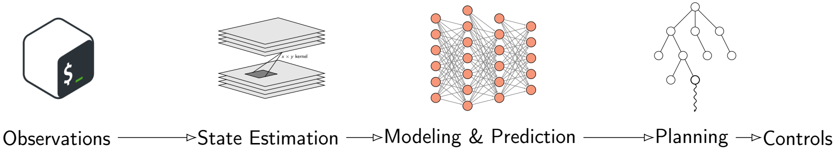
- Let $\mathcal{A} \triangleq \{\mathtt{nmap}_i, \mathtt{metasploit}_i, \mathtt{nikto}_i, \ldots\}$ be the **action space**. $\mathcal{A} \subset \mathcal{B}$ where $\mathcal{B}$ is the set of commands of the Bash command-line.
- Let

$$r(s_{t+1}|a_t, s_t) = \begin{cases} 10 & \text{if } b_{t+1}^{\#\text{flags}} > b_t^{\#\text{flags}} \\ 0 & \text{if } b_{t+1} \neq b_t \\ -10 & \text{otherwise} \end{cases}$$

be the **reward function**, realizing the agent's objective to capture the flags in the system.

System Model (3/3)

- As $|\mathcal{B}| >> 10^{34}$, we rely on parametric function approximation.
  - Consider parameterized policies π_θ
  - where $\theta \in \mathbb{R}^d \wedge d << 10^{34}$.
- We consider[28] the space of Non-Markovian **History-Dependent** Time-Homogeneous Mixed Policies $\pi : \mathcal{B} \mapsto \mathcal{A}, \quad \pi \in \Pi^{HR}$.

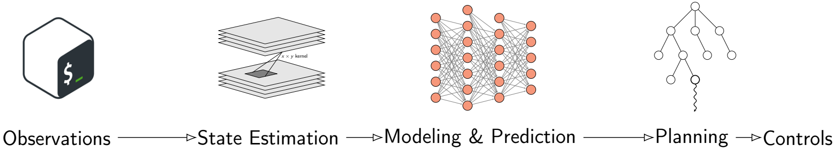


Observations → State Estimation → Modeling & Prediction → Planning → Controls

[28] $\mathcal{B}$ is the set of belief states.

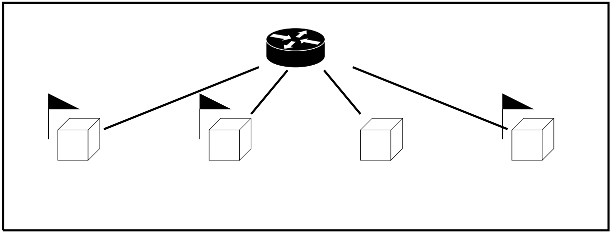
System Model (3/3)

- As $|\mathcal{B}| >> 10^{34}$, we rely on parameteric function approximation.
  - Consider parameterized policies π_θ
  - where $\theta \in \mathbb{R}^d \wedge d << 10^{34}$.
- We consider[29] the space of Non-Markovian **History-Dependent** Time-Homogeneous Mixed Policies $\pi : \mathcal{B} \mapsto \mathcal{A}, \quad \pi \in \Pi^{HR}$.



[29] $\mathcal{B}$ is the set of belief states.

First Task: v_1

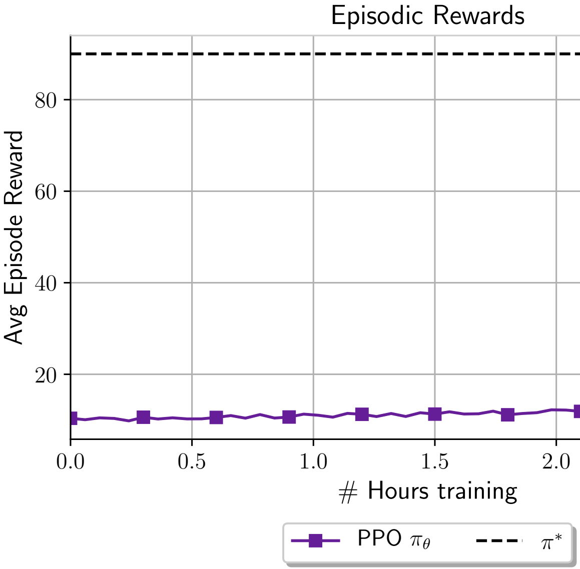


- **Goal:** Given no prior knowledge, except the IP of the subnetwork `172.18.1.0/24`, learn π^*_θ.

$$\pi^*_\theta = \mathop{\arg\max}_{\pi_\theta \in \Pi_\theta} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] \qquad \Pi_\theta = \{\pi_\theta \quad | \quad \theta \in \mathbb{R}^d\}$$

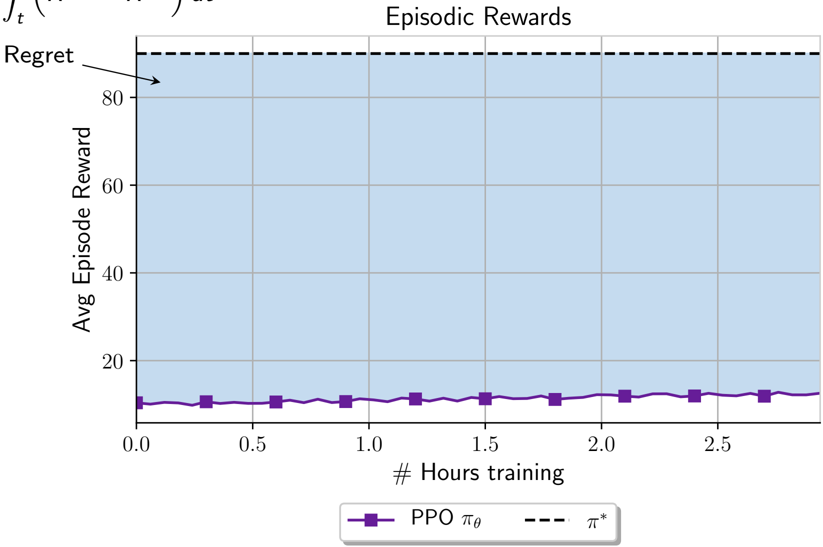
- π^*_θ: Finds all flags in the minimum number of steps

A First Attempt of the υ_1 Task!



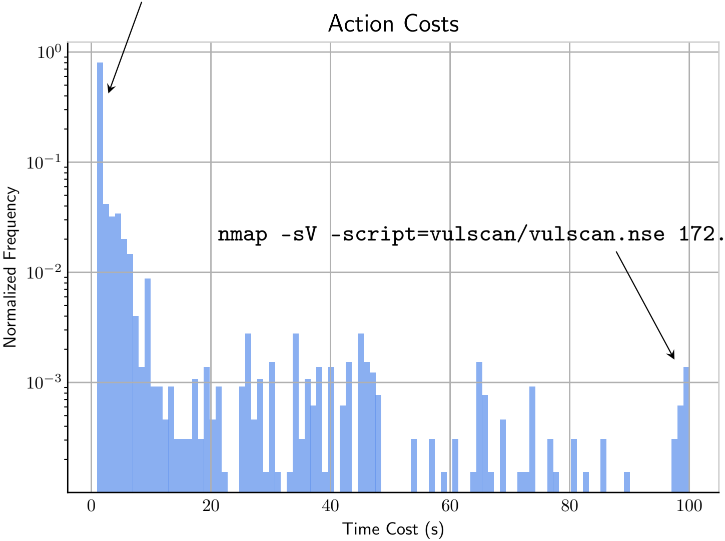
A First Attempt of the υ_1 Task!

$$\int_t \left(R^{\pi^*} - R^{\pi_\theta} \right) dt$$



Empirical Distribution of Action Costs

```
lftp ftp://u:pw@172.8.3.3
```



```
nmap -sV -script=vulscan/vulscan.nse 172.18.3.0/24
```

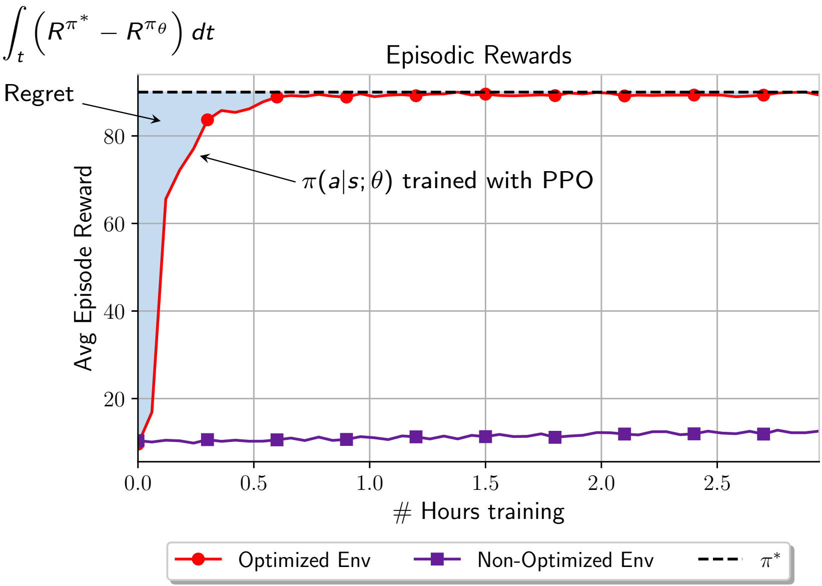
Some Assumptions

- With some loss of generality (**but not much**), we can assume a *Partially Synchronous System*
  - Access to **Eventually Perfect Failure Detector** $\Diamond\mathcal{P}$
    - (strong completeness and eventual strong accuracy)
  - **Eventual upper bounds** on communication delays
  - **Crash-stop failure model** extended with omission faults
- This system model enables optimizations:
  - Upper bound timeout on scanning operations
  - Scan results can be cached for some duration Δ
  - Pool SSH, Telnet, FTP, ... connections and re-use between episodes
  - Constrain action space per state s, $\mathcal{A}_s \subset \mathcal{A}$

Some Assumptions

- With some loss of generality (**but not much**), we can assume a *Partially Synchronous System*
  - Access to **Eventually Perfect Failure Detector** $\Diamond\mathcal{P}$
    - (strong completeness and eventual strong accuracy)
  - **Eventual upper bounds** on communication delays
  - **Crash-stop failure model** extended with omission faults
- This system model **enables optimizations**:
  - Upper bound timeout on scanning operations
  - Scan results can be cached for some duration Δ
  - Pool SSH, Telnet, FTP, ... connections and re-use between episodes
  - Constrain action space per state s, $\mathcal{A}_s \subset \mathcal{A}$

A **Second** Attempt of the υ_1 Task

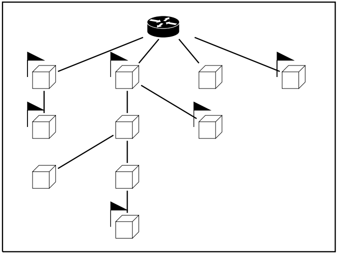


$$\int_t \left(R^{\pi^*} - R^{\pi_\theta}\right) dt$$

Regret

$\pi(a|s;\theta)$ trained with PPO

Second Task: υ_2



Learning task υ_2

- **Goal:** Given no prior knowledge, except the IP of the subnetwork `172.18.1.0/24`, learn π^*_θ.

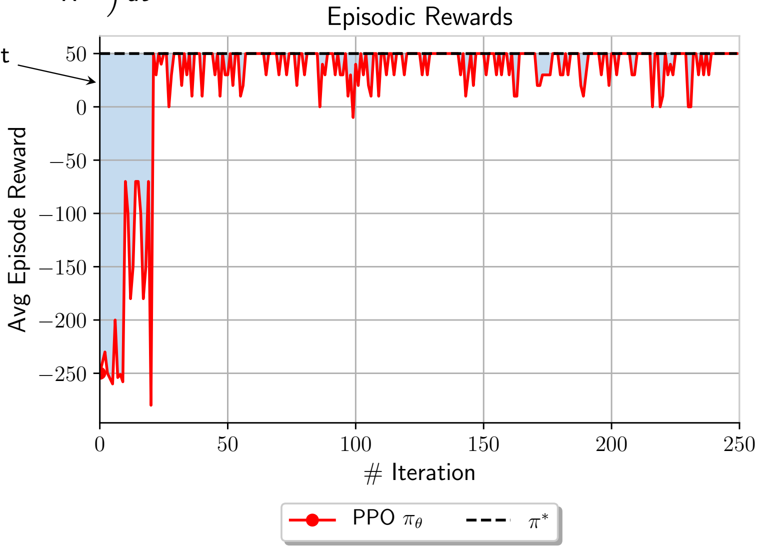
$$\pi^*_\theta = \underset{\pi_\theta \in \Pi_\theta}{\arg\max}\, \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] \qquad \Pi_\theta = \{\pi_\theta \quad | \quad \theta \in \mathbb{R}^d\}$$

- π^*_θ: Finds all flags in the minimum number of steps

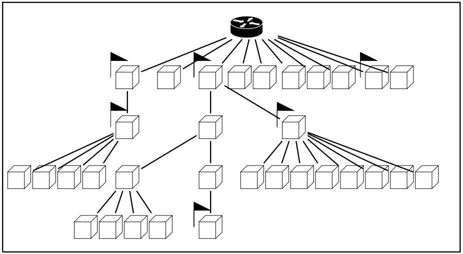
υ_2 Task Training Results

$$\int_t \left(R^{\pi^*} - R^{\pi_\theta}\right) dt$$

Regret



Third Task: υ_3



- **Goal:** Given no prior knowledge, except the IP of the subnetwork `172.18.1.0/24`, learn π^*_θ.

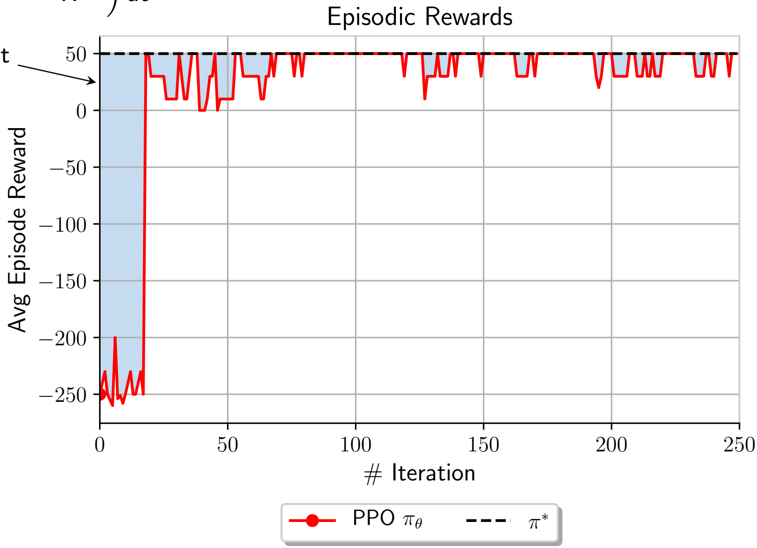
$$\pi^*_\theta = \underset{\pi_\theta \in \Pi_\theta}{\arg\max}\, \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1}\right] \qquad \Pi_\theta = \{\pi_\theta \quad | \quad \theta \in \mathbb{R}^d\}$$

- π^*_θ: Finds all flags in the minimum number of steps

v_3 Task Training Results

$$\int_t \left(R^{\pi^*} - R^{\pi_\theta}\right) dt$$

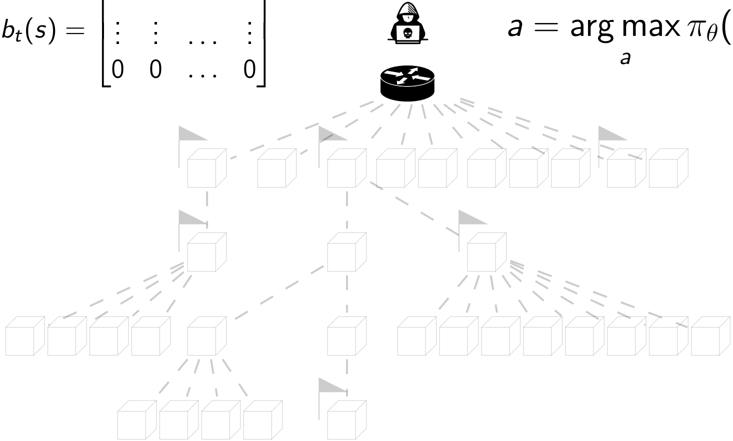
Regret



Example of a Learned Policy π_θ

$$b_t(s) = \begin{bmatrix} 0 & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$a = \arg\max_a \pi_\theta(a|s)$$



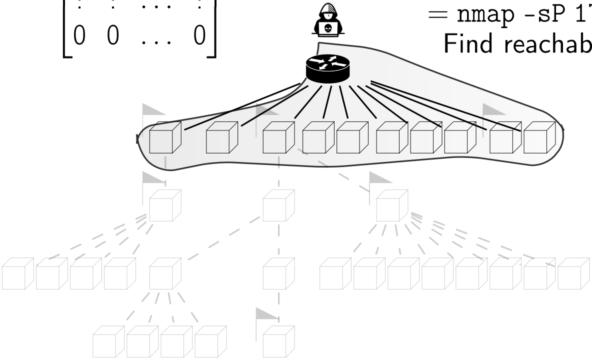
Example of a Learned Policy π_θ

$$b_t = \begin{bmatrix} \color{red}{1} & 0 & \dots & 0 \\ \color{red}{1} & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$a_t = \arg\max_a \pi_\theta(a|b_t)$$

$= \texttt{nmap -sP 172.18.3.0/24}$

Find reachable machines on network

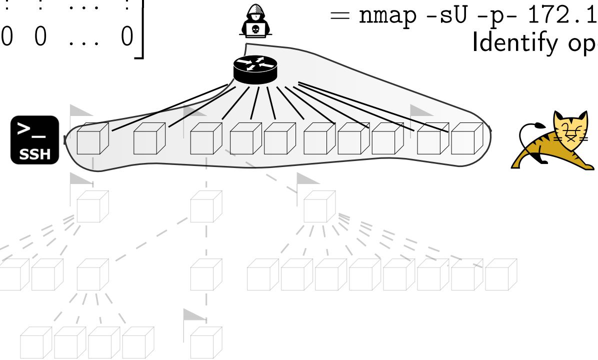


Example of a Learned Policy π_θ

$$b_t = \begin{bmatrix} 1 & 1 & \dots & 0 \\ 1 & 2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\begin{aligned} a_t &= \arg\max_a \pi_\theta(a|b_t) \\ &= \texttt{nmap -sU -p- 172.18.3.0/24} \end{aligned}$$

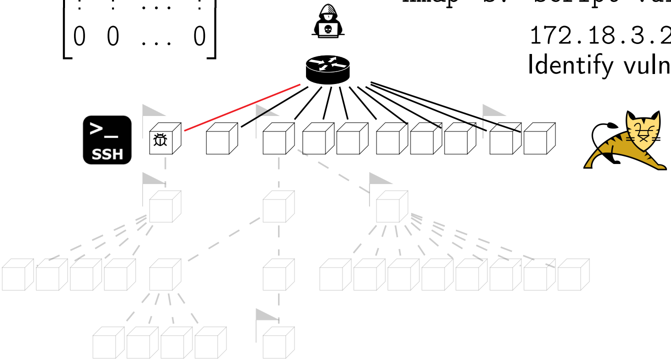
Identify open ports



Example of a Learned Policy π_θ

$$b_t = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\begin{aligned} a_t &= \arg\max_a \pi_\theta(a|b_t) \\ &= \texttt{nmap -sV -script=vulscan/vulscan.nse} \end{aligned}$$



172.18.3.2
Identify vulnerabilities

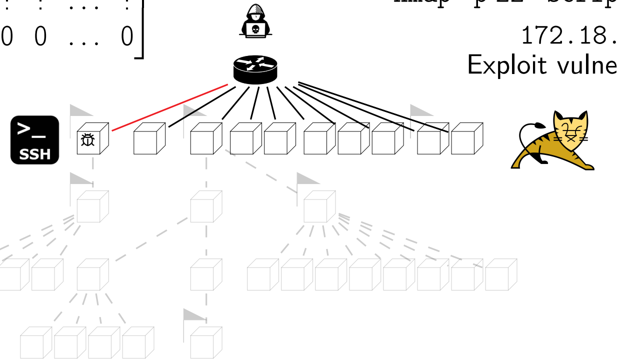
Example of a Learned Policy π_θ

$$b_t = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\begin{aligned} a_t &= \arg\max_a \pi_\theta(a|b_t) \\ &= \texttt{nmap -p 22 -script ssh-brute} \end{aligned}$$

`172.18.3.2`

Exploit vulnerability

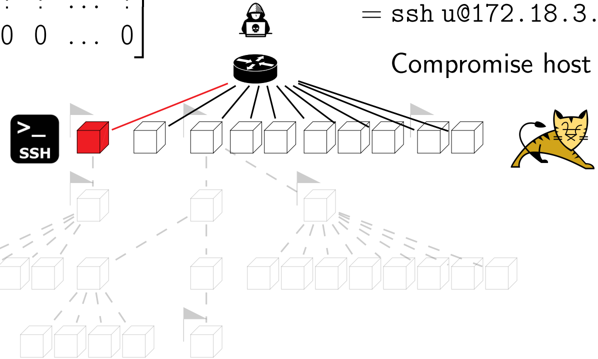


Example of a Learned Policy π_θ

$$b_t = \begin{bmatrix} 1 & 1 & \dots & \color{red}{1} \\ 1 & 2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\begin{aligned} a_t &= \arg\max_a \pi_\theta(a|b_t) \\ &= \texttt{ssh u@172.18.3.2} \end{aligned}$$

Compromise host

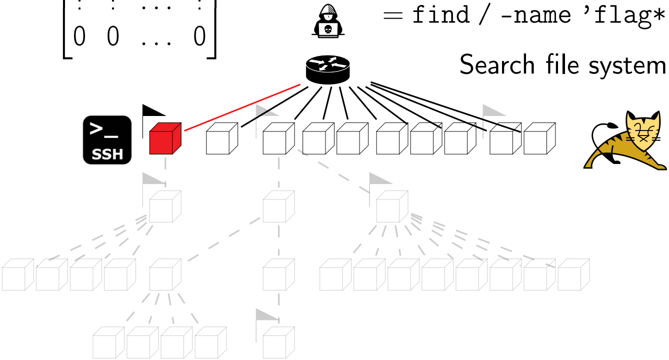


Example of a Learned Policy π_θ

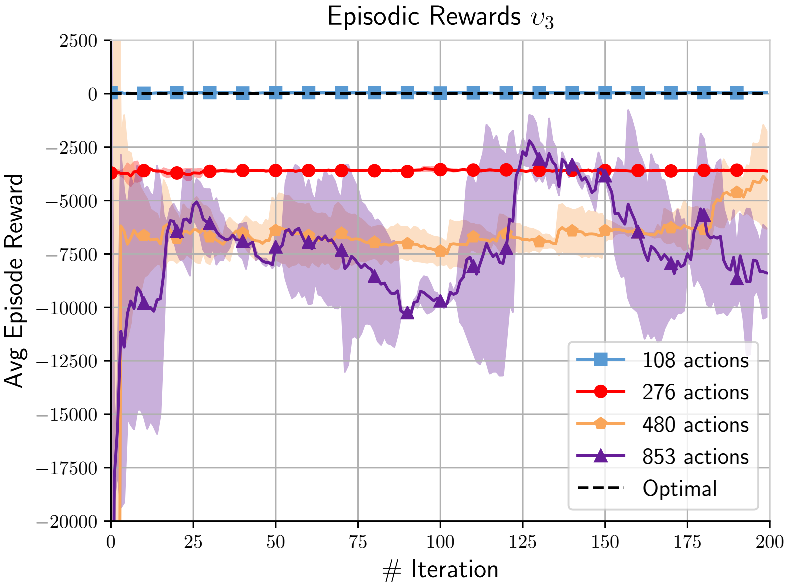
$$b_t = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 2 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\begin{aligned} a_t &= \arg\max_a \pi_\theta(a|b_t) \\ &= \texttt{find / -name 'flag*.txt'} \end{aligned}$$

Search file system

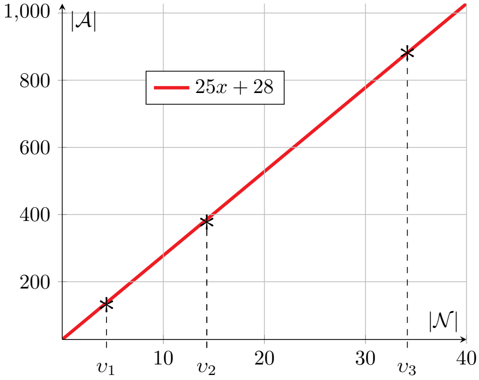


Challenge: Learning with Large Action Spaces



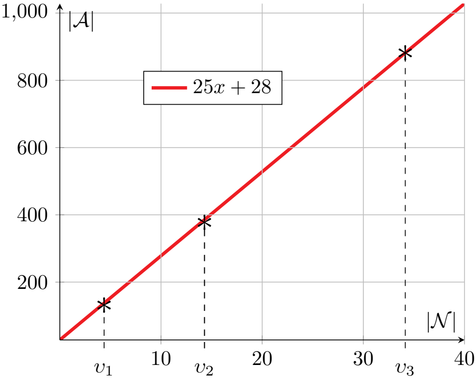
Action Space Scaling

- Actions **scale linearly** with the number of nodes $|\mathcal{N}|$.
- $|\mathcal{A}| = \underbrace{25|\mathcal{N}|}_{\text{Node-actions}} + \underbrace{28}_{\text{Subnet actions}}$
- Large action spaces is a **known challenge for RL**
- **Reason**: Inflates the policy space Π exponentially
    - Assume Markovian Deterministic Stationary policies
    - $|\Pi| = (|\mathcal{S}|^{|\mathcal{A}|})^{T-1} = ((|\mathcal{S}|)^{|\mathcal{A}|})^{T-1}$



Action Space Scaling

- Actions scale linearly with the number of nodes $|\mathcal{N}|$.
- $|\mathcal{A}| = \underbrace{25|\mathcal{N}|}_{\text{Node-actions}} + \underbrace{28}_{\text{Subnet actions}}$
- Large action spaces is a **known challenge for RL**
- **Reason**: Inflates the policy space Π exponentially
  - Assume Markovian Deterministic Stationary policies
  - $|\Pi| = \left(|\mathcal{S}|^{|\mathcal{A}|}\right)^{T-1} = ((|\mathcal{S}|)^{|\mathcal{A}|})^{T-1}$

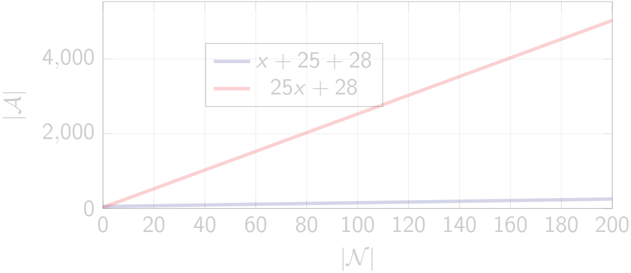


A Possible Solution: Auto-Regressive Policy

- **Idea:** Represent action a_t as *sequence of sub-actions* (n, a):
  1. **Select node** in the topology to target (n)
  2. **Select action** to apply to node (a)
- Then, $\pi(a|o) = \pi(a, n|o)$, which can be decomposed into $\pi(a|n, o)$ & $\pi(n|o)$:

$$\pi(a, n|o) = \pi(a|n, o) \cdot \pi(n|o) \qquad \text{Chain rule}$$

- Reduces the size of the action space from $25|\mathcal{N}| + 28$ to $|\mathcal{N}| + 25 + 28$
  - Still $\mathcal{O}(n)$, though.

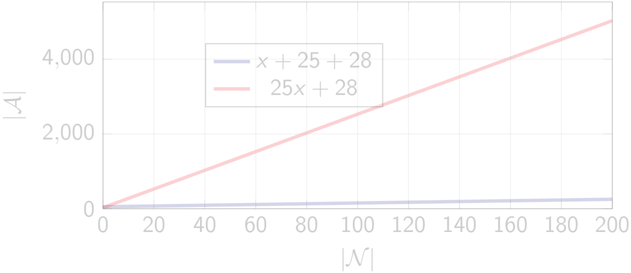


A Possible Solution: Auto-Regressive Policy

- **Idea:** Represent action a_t as *sequence of sub-actions* (n, a):
  1. **Select node** in the topology to target (n)
  2. **Select action** to apply to node (a)
- Then, $\pi(a|o) = \pi(a, n|o)$, which can be decomposed into $\pi(a|n, o)$ & $\pi(n|o)$:

$$\pi(a, n|o) = \pi(a|n, o) \cdot \pi(n|o) \qquad \text{Chain rule}$$

- Reduces the size of the action space from $25|\mathcal{N}| + 28$ to $|\mathcal{N}| + 25 + 28$
  - Still $\mathcal{O}(n)$, though.

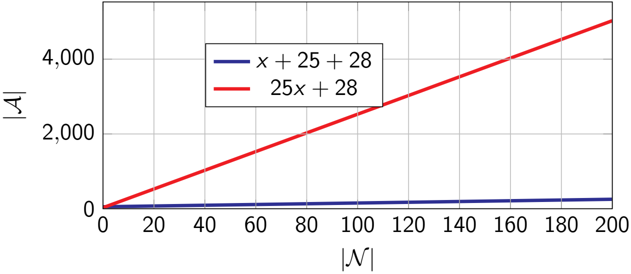


A Possible Solution: Auto-Regressive Policy

- **Idea:** Represent action a_t as *sequence of sub-actions* (n, a):
  1. **Select node** in the topology to target (n)
  2. **Select action** to apply to node (a)
- Then, $\pi(a|o) = \pi(a, n|o)$, which can be decomposed into $\pi(a|n, o)$ & $\pi(n|o)$:

$$\pi(a, n|o) = \pi(a|n, o) \cdot \pi(n|o) \qquad \text{Chain rule}$$

- Reduces the size of the action space from $25|\mathcal{N}| + 28$ to $|\mathcal{N}| + 25 + 28$
  - Still $\mathcal{O}(n)$, though.



Conclusions and Future Work

- It is challenging to use decision-theoretic methods for controlling complex systems
  - Simulation/Abstract Models are effective to deal with scale, but...
  - We also want to **ensure grounding in real world applications**

- We investigate how to combine real security applications with decision theory/learning theory methods.
  - Propose a framework/system for learning control-tasks in security
  - Shown on simple tasks that the approach is feasible
  - **We seek the right trade-off between real-system interaction and simulation/models**—Open research question

- **Future work**:
  - Many challenges remain..
  - Domain randomization and generalization
  - More elaborate learning tasks
  - Model-based RL

Conclusions and Future Work

- It is challenging to use decision-theoretic methods for controlling complex systems
  - Simulation/Abstract Models are effective to deal with scale, but...
  - We also want to **ensure grounding in real world applications**
- We investigate how to combine real security applications with decision theory/learning theory methods.
  - Propose a framework/system for learning control-tasks in security
  - Shown on simple tasks that the approach is feasible
  - **We seek the right trade-off between real-system interaction and simulation/models**—Open research question
- **Future work**:
  - Many challenges remain..
  - Domain randomization and generalization
  - More elaborate learning tasks
  - Model-based RL

Conclusions and Future Work

- It is challenging to use decision-theoretic methods for controlling complex systems
  - Simulation/Abstract Models are effective to deal with scale, but...
  - We also want to **ensure grounding in real world applications**

- We investigate how to combine real security applications with decision theory/learning theory methods.
  - Propose a framework/system for learning control-tasks in security
  - Shown on simple tasks that the approach is feasible
  - **We seek the right trade-off between real-system interaction and simulation/models**—Open research question

- **Future work**:
  - Many challenges remain..
  - Domain randomization and generalization
  - More elaborate learning tasks
  - Model-based RL