# Automated Intrusion Response

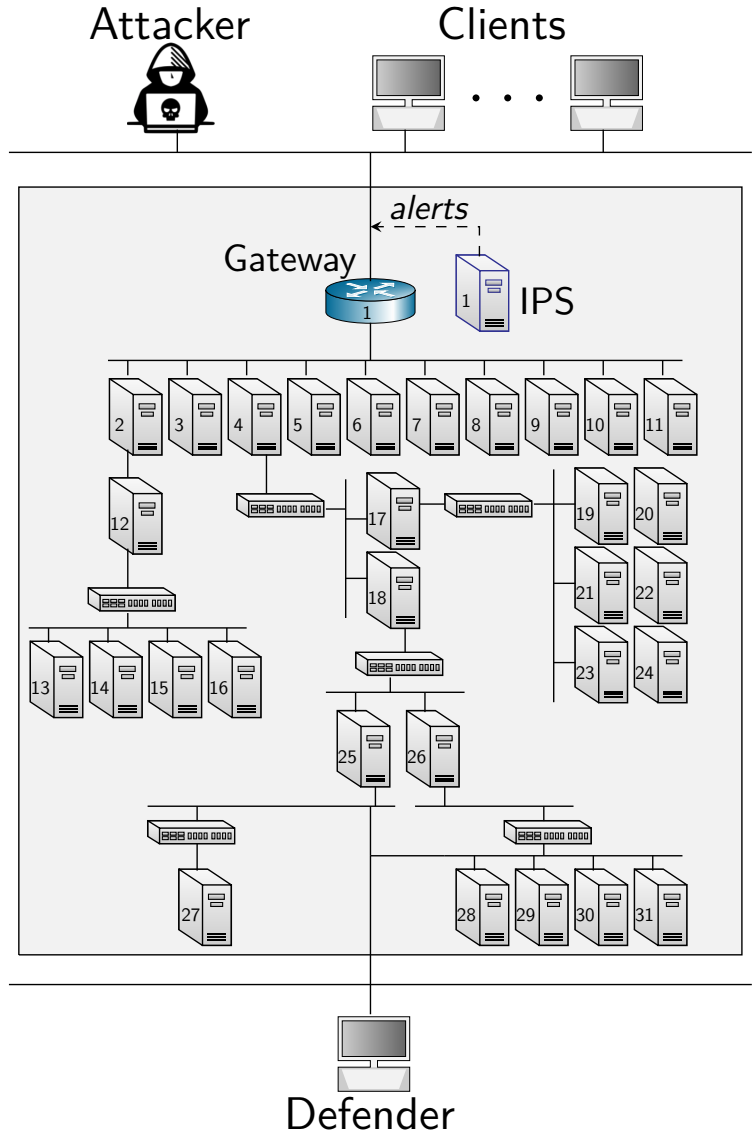
Kim Hammar

*kimham@kth.se*
Division of Network and Systems Engineering
KTH Royal Institute of Technology

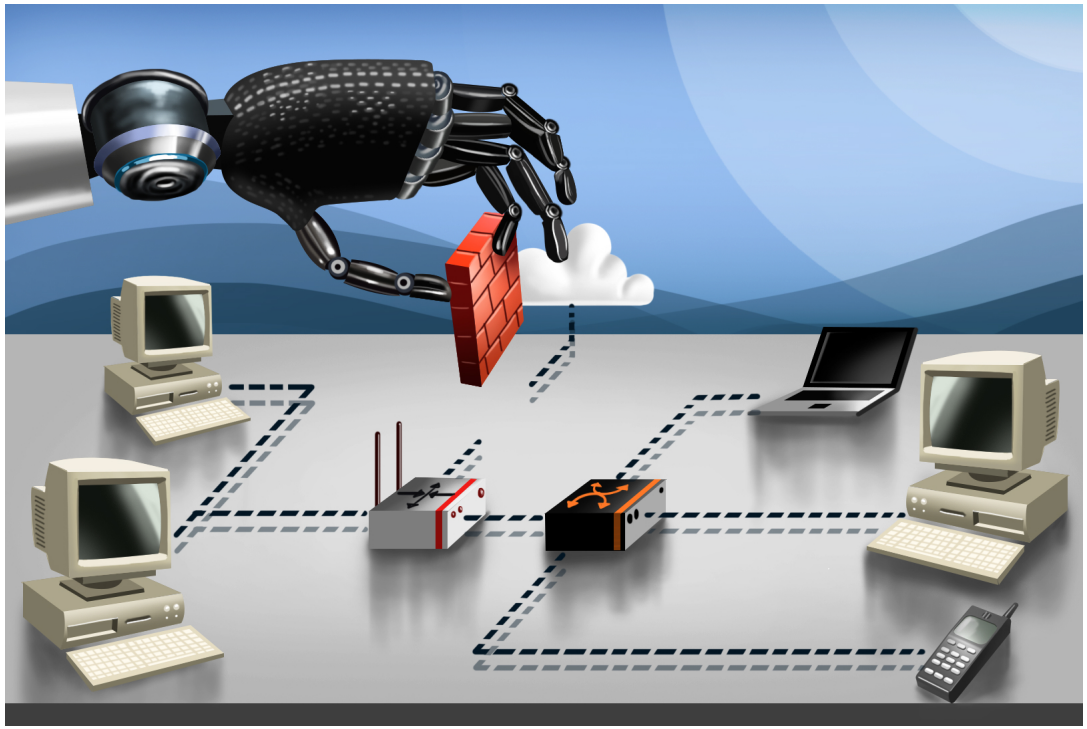
May 22, 2024

# Use Case: Intrusion Response

- A **defender** owns an infrastructure
  - Consists of connected components
  - Components run network services
  - Defender defends the infrastructure by monitoring and active defense
  - Has partial observability
- An **attacker** seeks to intrude on the infrastructure
  - Has a partial view of the infrastructure
  - Wants to compromise specific components
  - Attacks by reconnaissance, exploitation and pivoting



# Automated Intrusion Response



**Levels of security automation**

***No automation.***
Manual detection.
Manual prevention.
Lack of tools.

1980s

***Operator assistance.***
Audit logs
Manual detection.
Manual prevention.

1990s

***Partial automation.***
Manual configuration.
Intrusion detection systems.
Intrusion prevention systems.

2000s-Now



***High automation.***
System automatically updates itself.

Research

# Automated Intrusion Response

Can we find effective security strategies through decision-theoretic methods?

**Levels of security automation**

***No automation.***
Manual detection.
Manual prevention.
Lack of tools.

1980s

***Operator assistance.***
Audit logs
Manual detection.
Manual prevention.

1990s

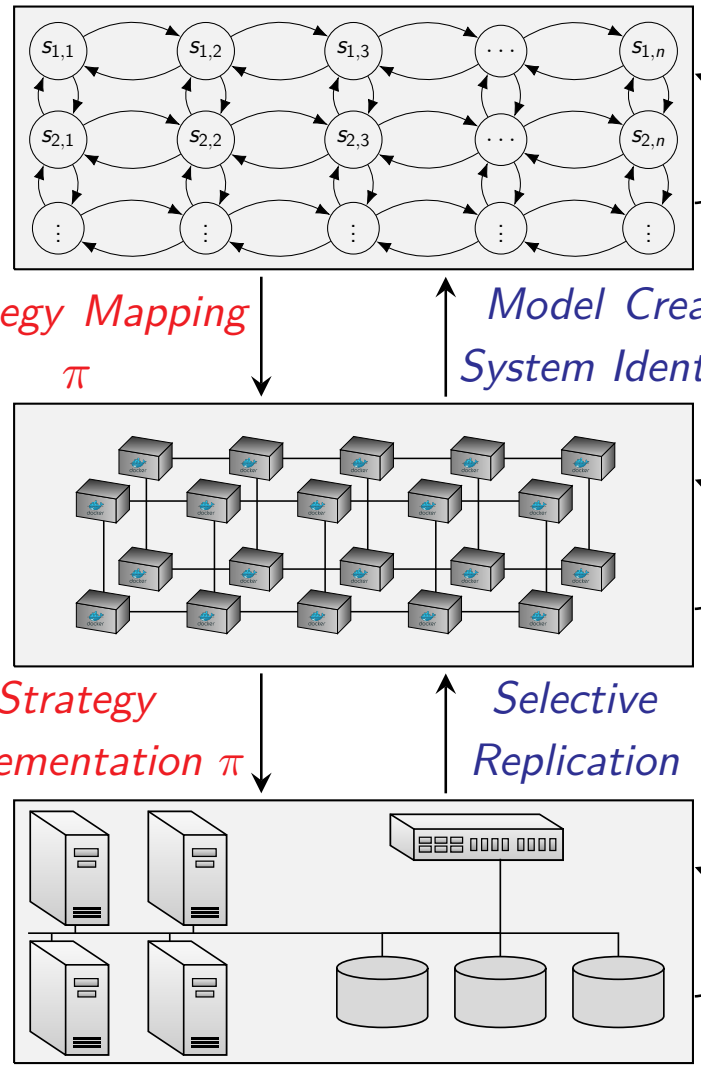
***Partial automation.***
Manual configuration.
Intrusion detection systems.
Intrusion prevention systems.

2000s-Now

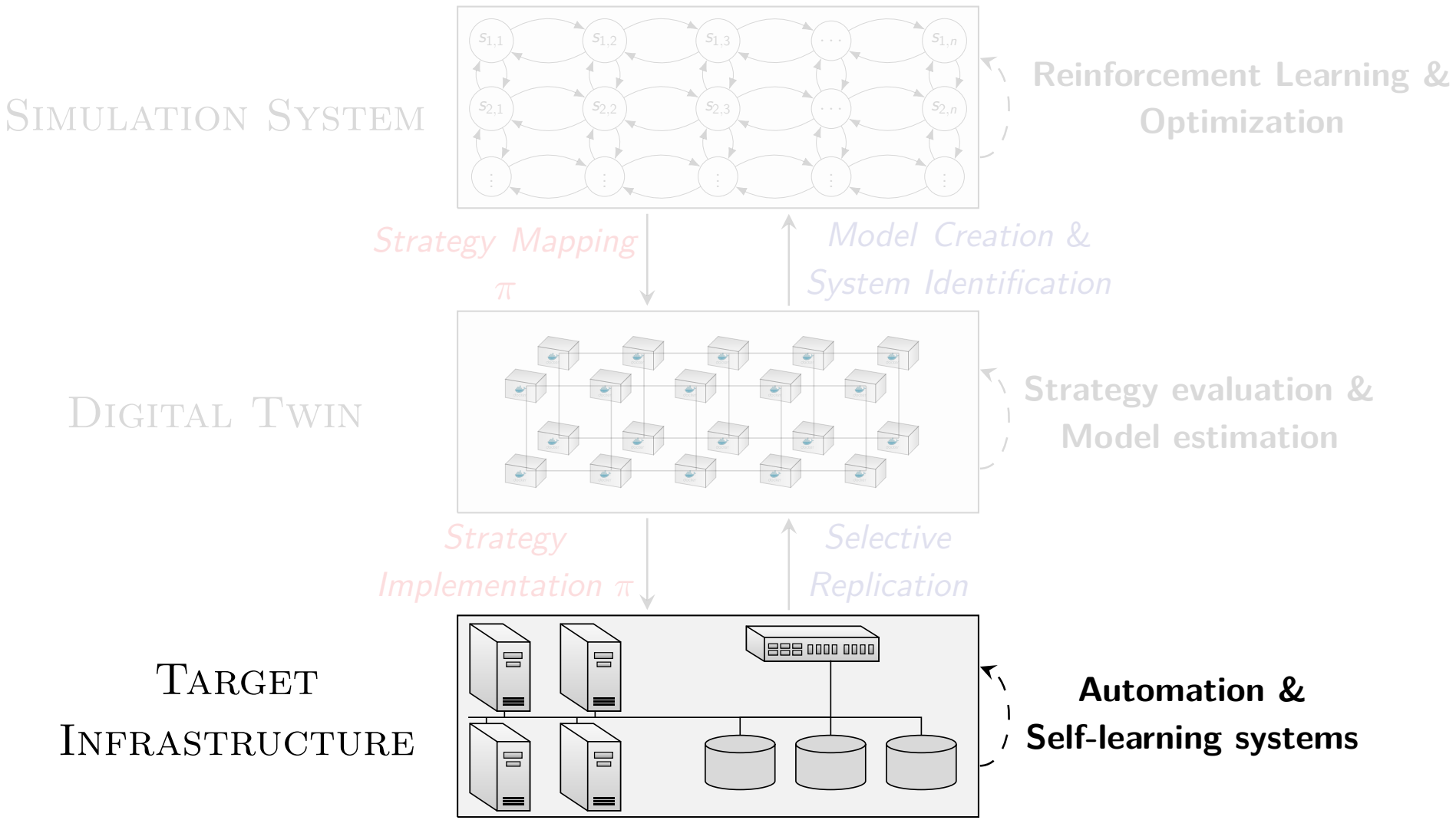
***High automation.***
System automatically
updates itself.

Research

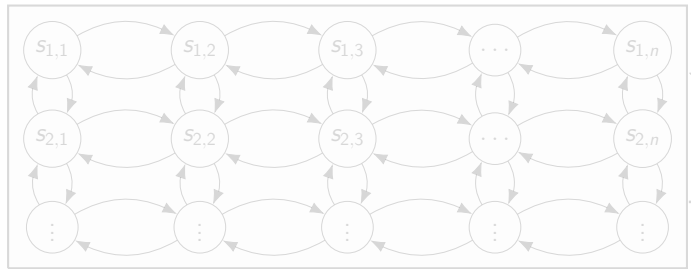
# Our Framework for Automated Intrusion Response



# Our Framework for Automated Intrusion Response



# Our Framework for Automated Intrusion Response

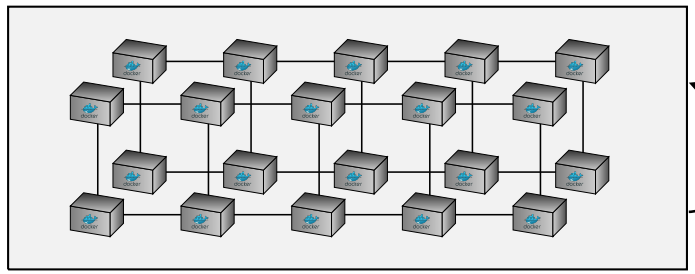



Simulation System

Reinforcement Learning & Optimization

*Strategy Mapping* $\pi$

*Model Creation & System Identification*

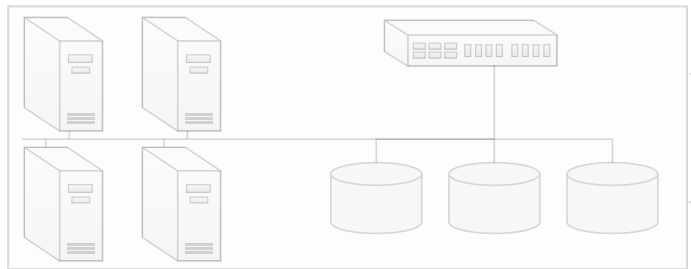


Digital Twin

**Model estimation**

*Strategy Implementation* $\pi$

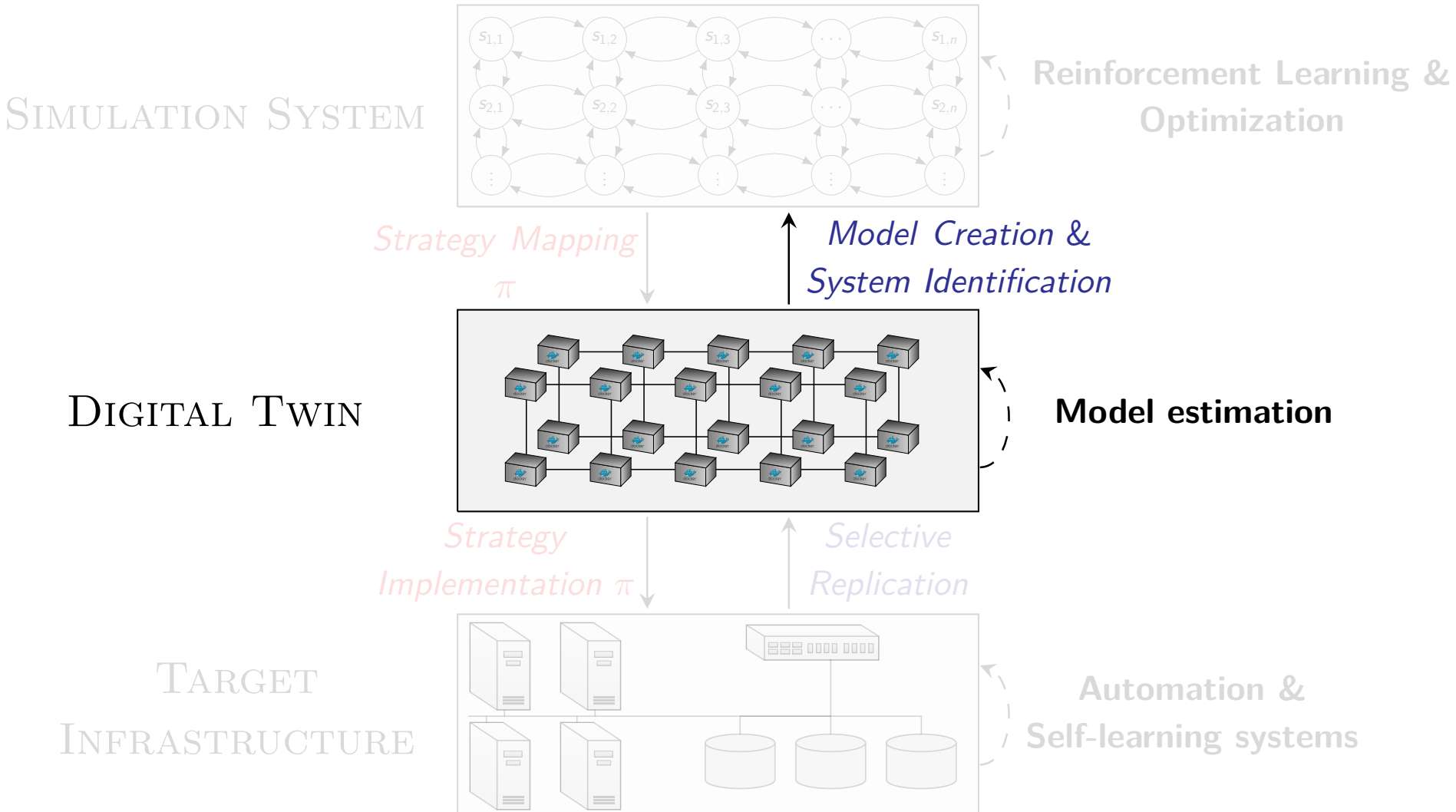
*Selective Replication*



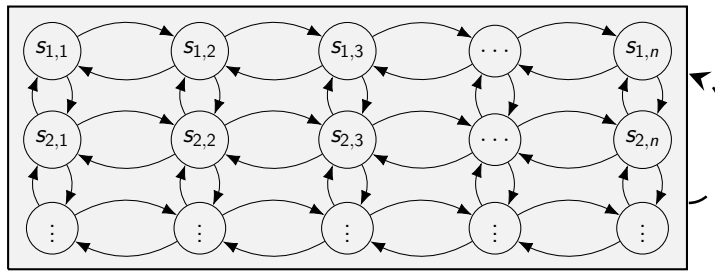
Target Infrastructure

Automation & Self-learning systems

# Our Framework for Automated Intrusion Response



# Our Framework for Automated Intrusion Response

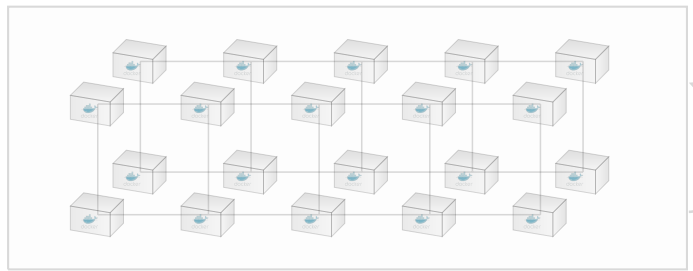



Simulation System

**Reinforcement Learning & Optimization**

*Strategy Mapping* $\pi$

*Model Creation & System Identification*

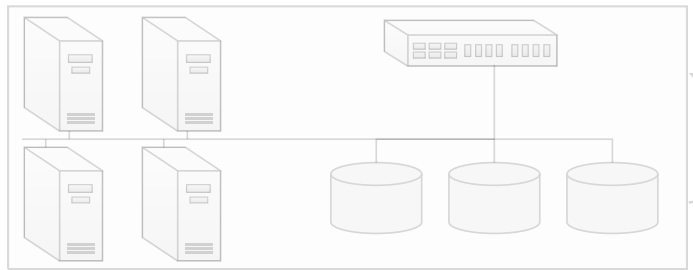


Digital Twin

Strategy evaluation & Model estimation

*Strategy Implementation* $\pi$

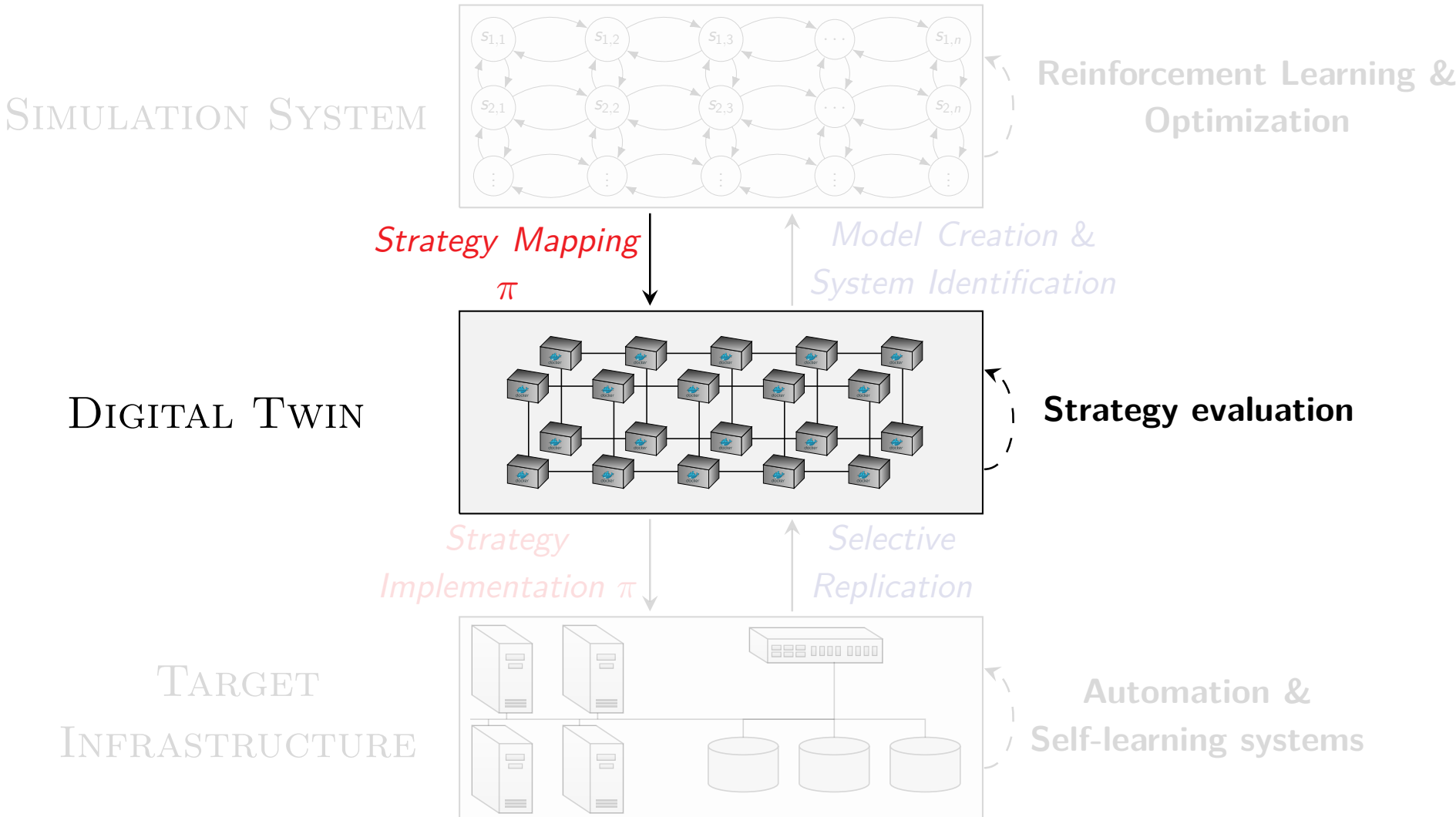
*Selective Replication*



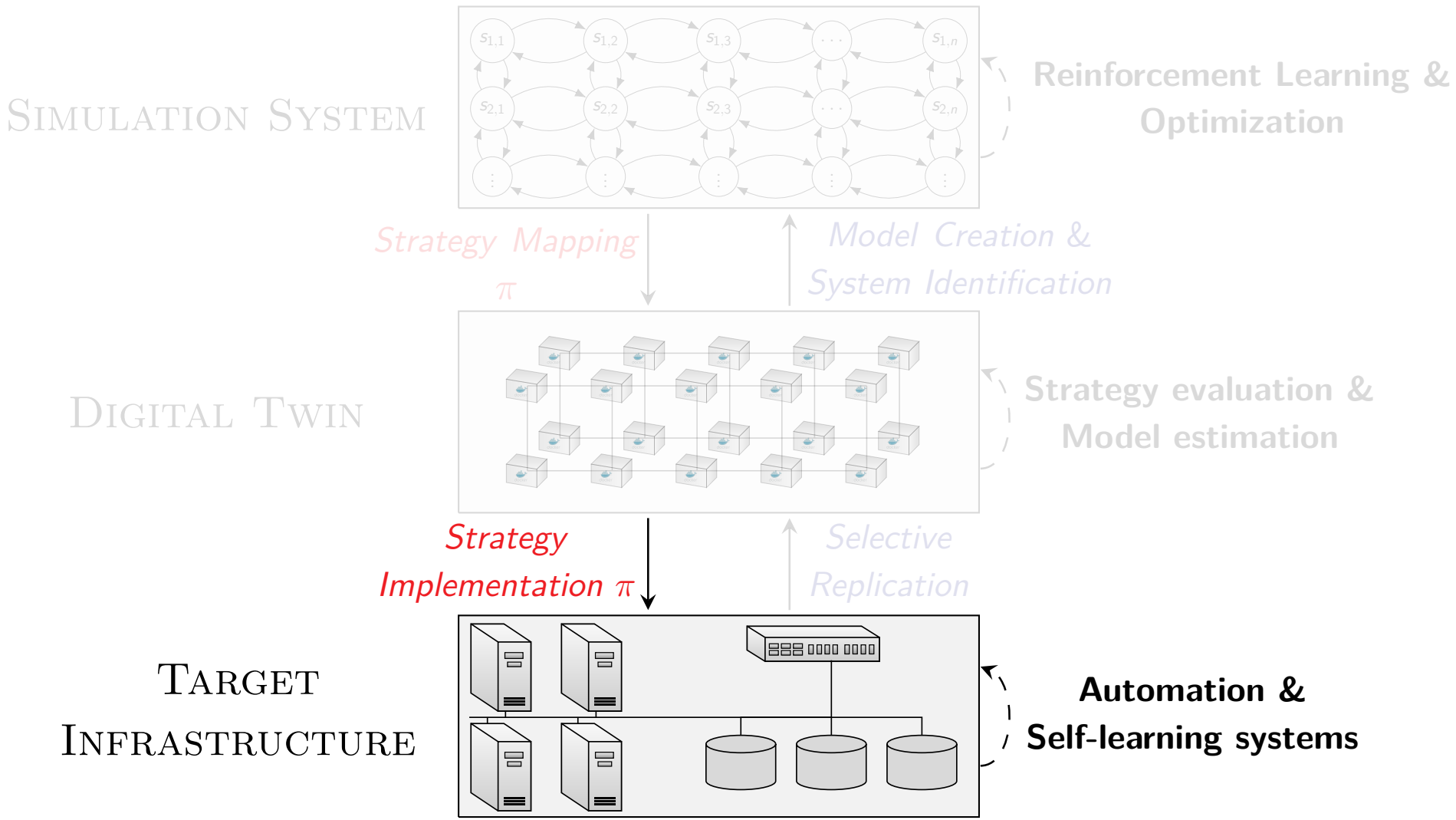
Target Infrastructure

Automation & Self-learning systems

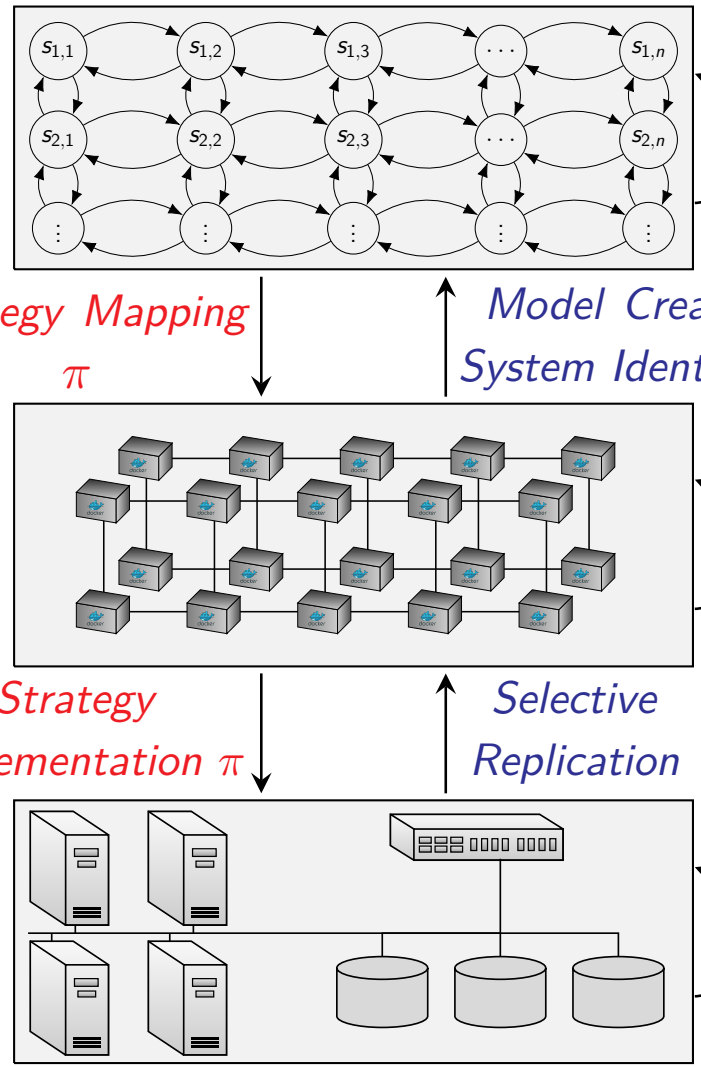
# Our Framework for Automated Intrusion Response



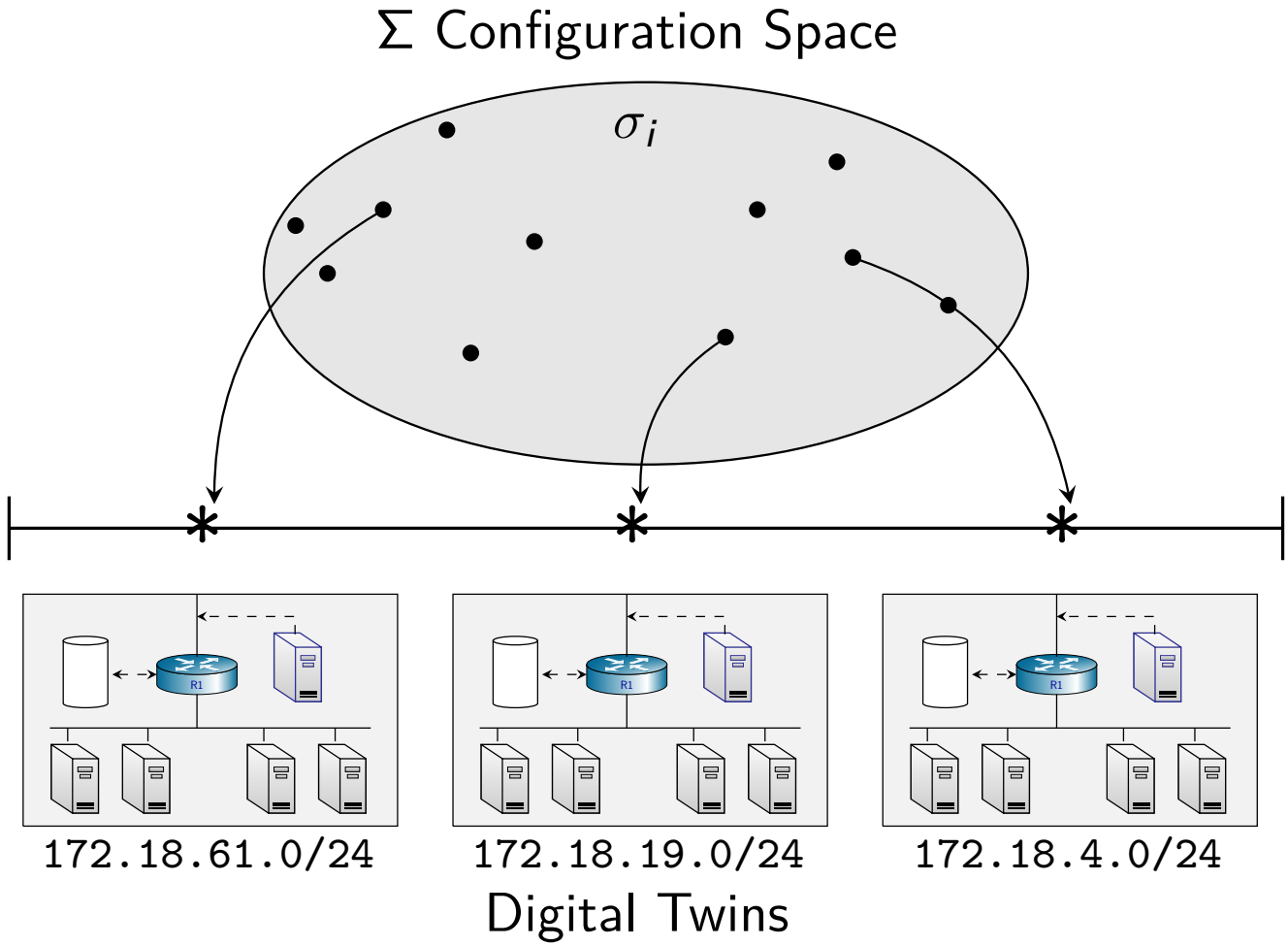
# Our Framework for Automated Intrusion Response



# Our Framework for Automated Intrusion Response



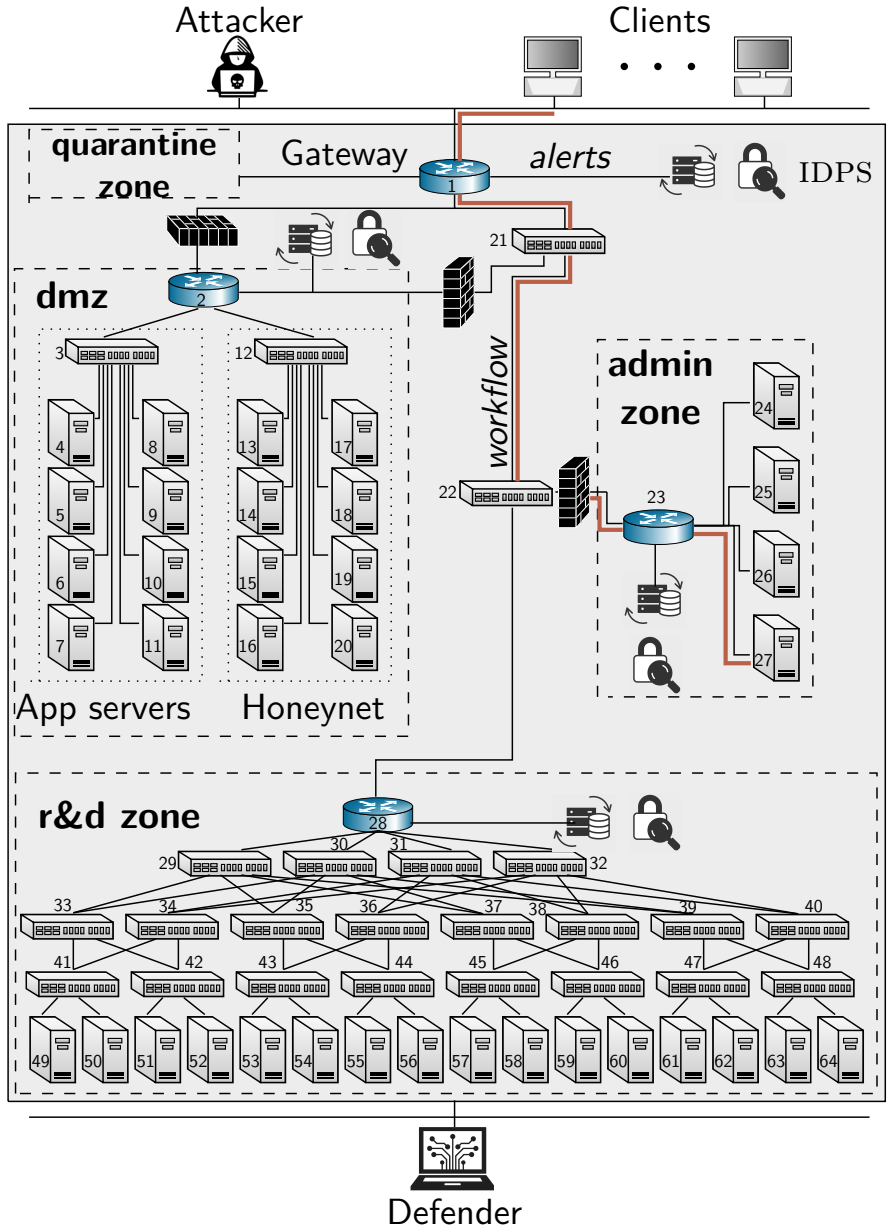
# Creating a Digital Twin of the Target Infrastructure



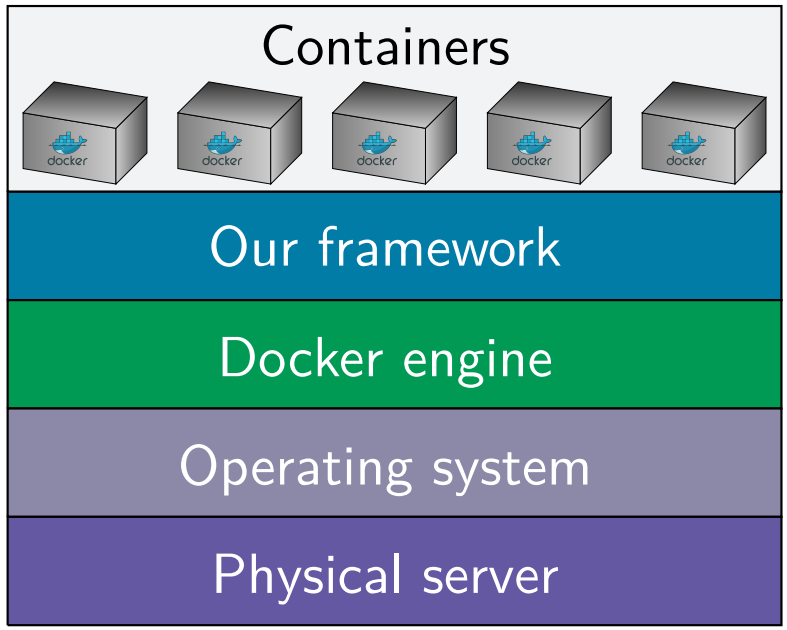

- Given an **infrastructure configuration**, our framework automates the creation of a digital twin.
- The configuration space defines the class of infrastructures that we can emulate.

# Example Infrastructure Configuration

- 64 **nodes**
  - 24 OVS switches
  - 3 gateways
  - 6 honeypots
  - 8 application servers
  - 4 administration servers
  - 15 compute servers
- 11 **vulnerabilities**
  - CVE-2010-0426
  - CVE-2015-3306
  - etc.
- **Management**
  - 1 SDN controller
  - 1 Kafka server
  - 1 elastic server

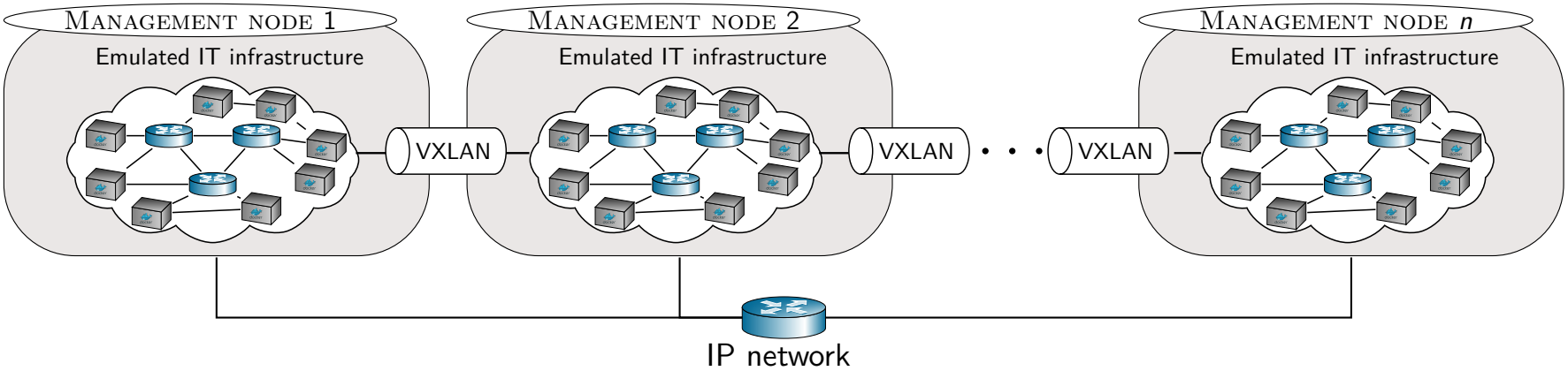


# Emulating Physical Components




- We emulate physical components with **Docker containers**
- Focus on linux-based systems
- Our framework provides the orchestration layer

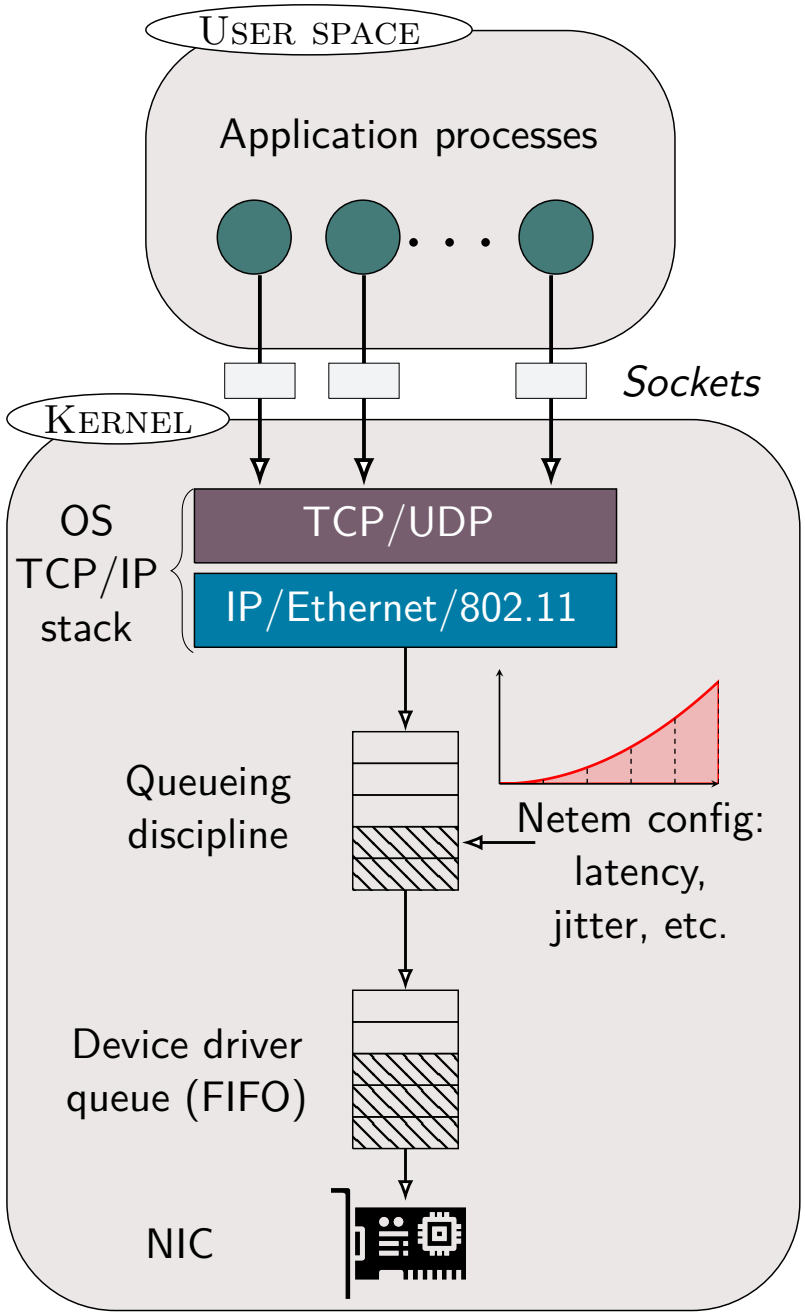
# Emulating Network Connectivity



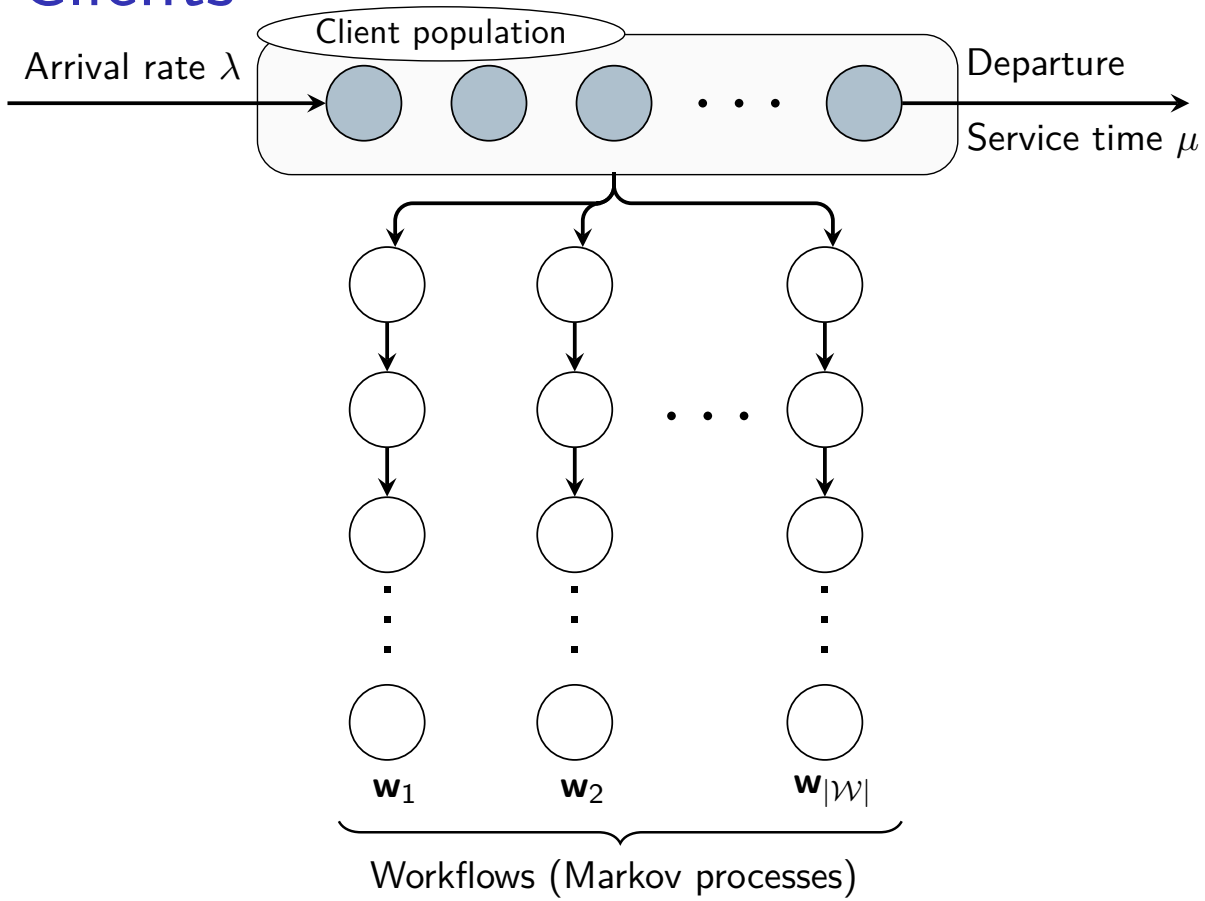

- We emulate network connectivity on the same host using **network namespaces**
- Connectivity across physical hosts is achieved using **VXLAN tunnels** with Docker swarm

# Emulating Network Conditions

- Traffic shaping using NetEm
- Allows to configure:
  - Delay
  - Capacity
  - Packet Loss
  - Jitter
  - Queueing delays
  - etc.



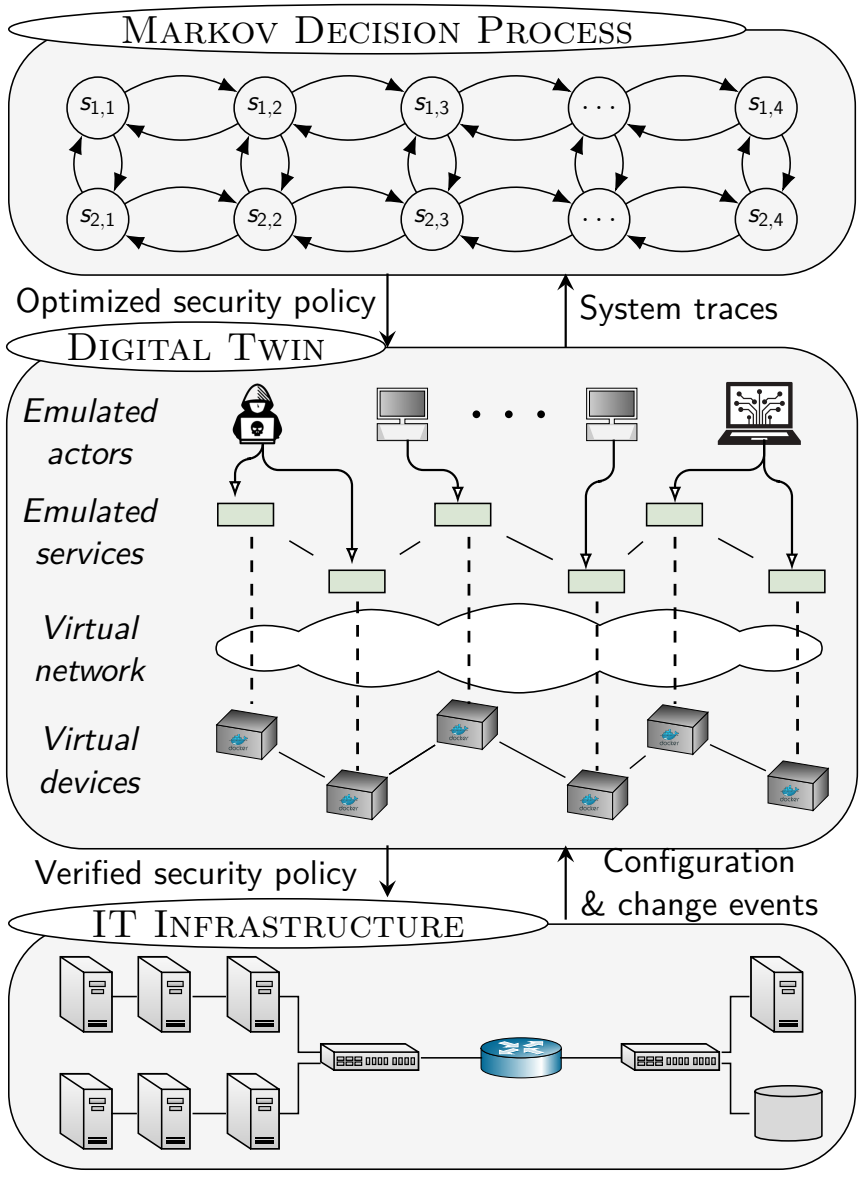
# Emulating Clients



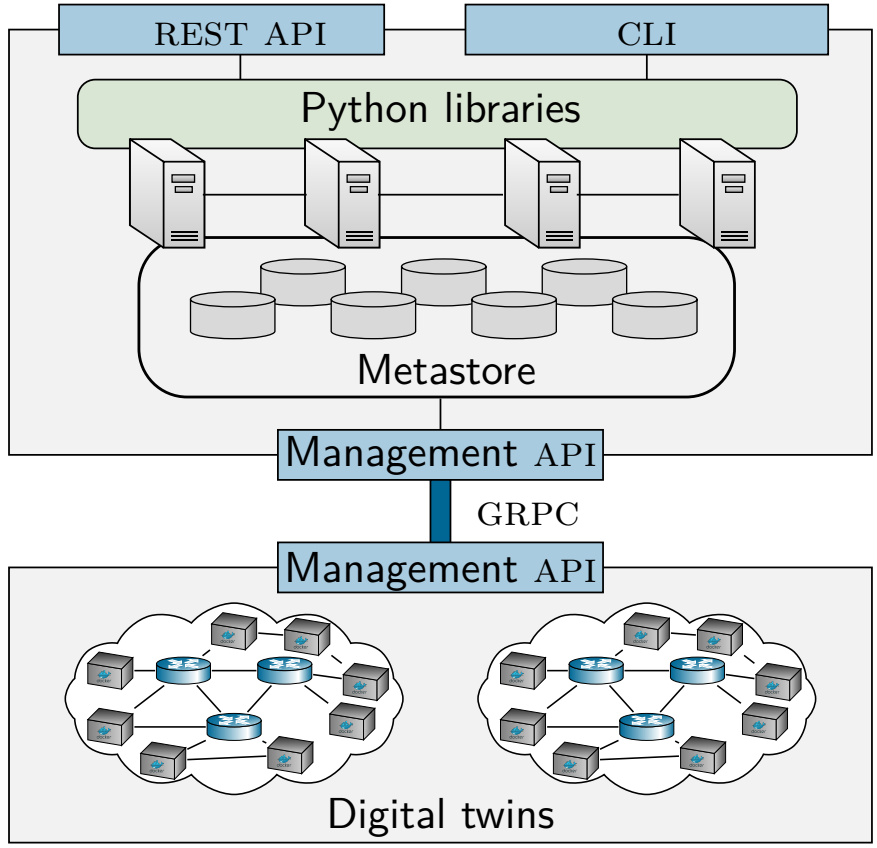

- Homogeneous client population
- Clients arrive according to $Po(\lambda)$
- Client service times $Exp(\mu)$
- Service dependencies $(S_t)_{t=1,2,\ldots} \sim \mathrm{MC}$

# Emulating The Attacker and The Defender

- **API for automated defender and attacker actions**
- **Attacker actions:**
  - Exploits
  - Reconnaissance
  - Pivoting
  - etc.
- **Defender actions:**
  - Shut downs
  - Redirect
  - Isolate
  - Recover
  - Migrate
  - etc.

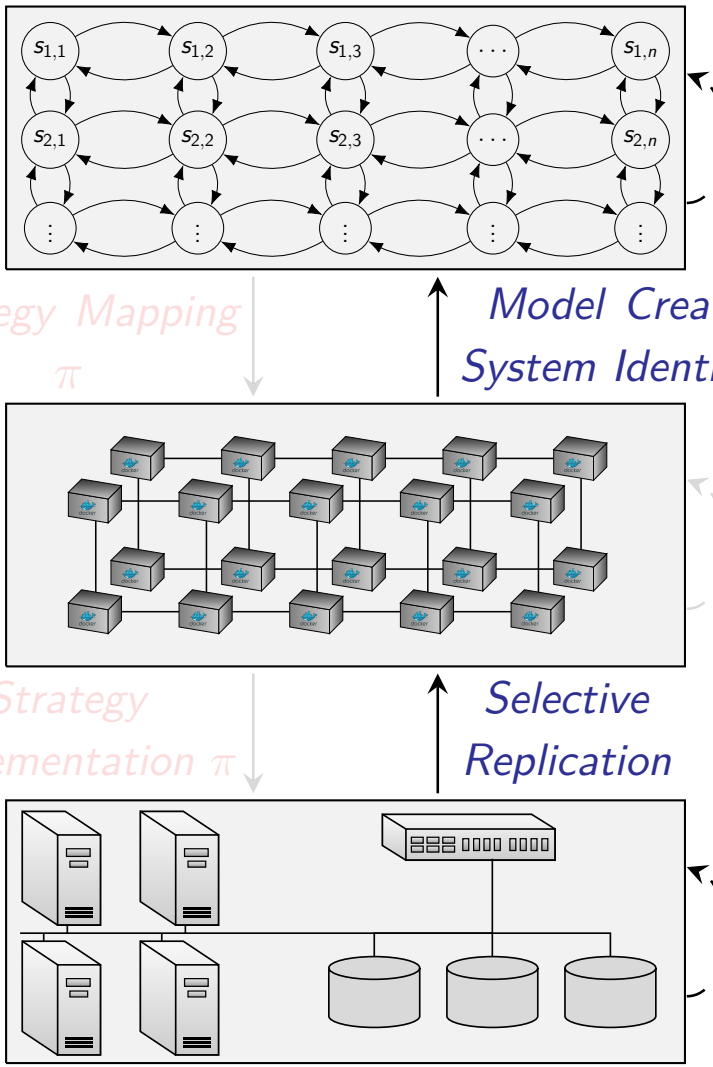


# Software framework



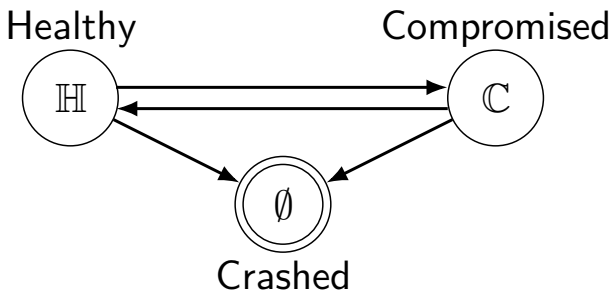

- More details about the software framework
  - Source code: https://github.com/Limmen/csle
  - Documentation: http://limmen.dev/csle/
  - Demo: https://www.youtube.com/watch?v=iE2KPmtIs2A
  - Installation: https://www.youtube.com/watch?v=l_g3sRJwwhc

# System Identification

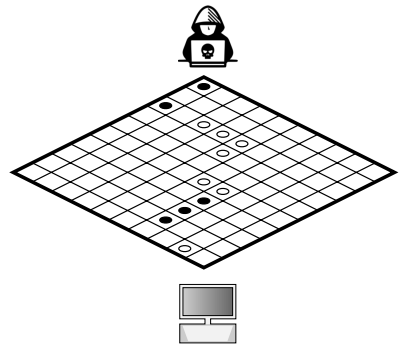


# System Model

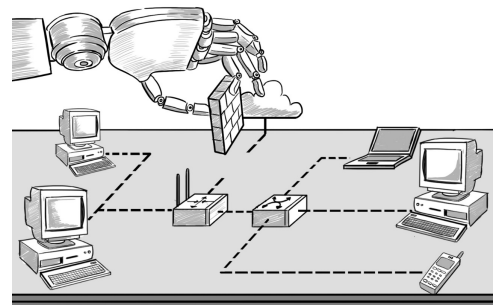
Static attacker
Small set of responses




Dynamic attacker
Small set of responses



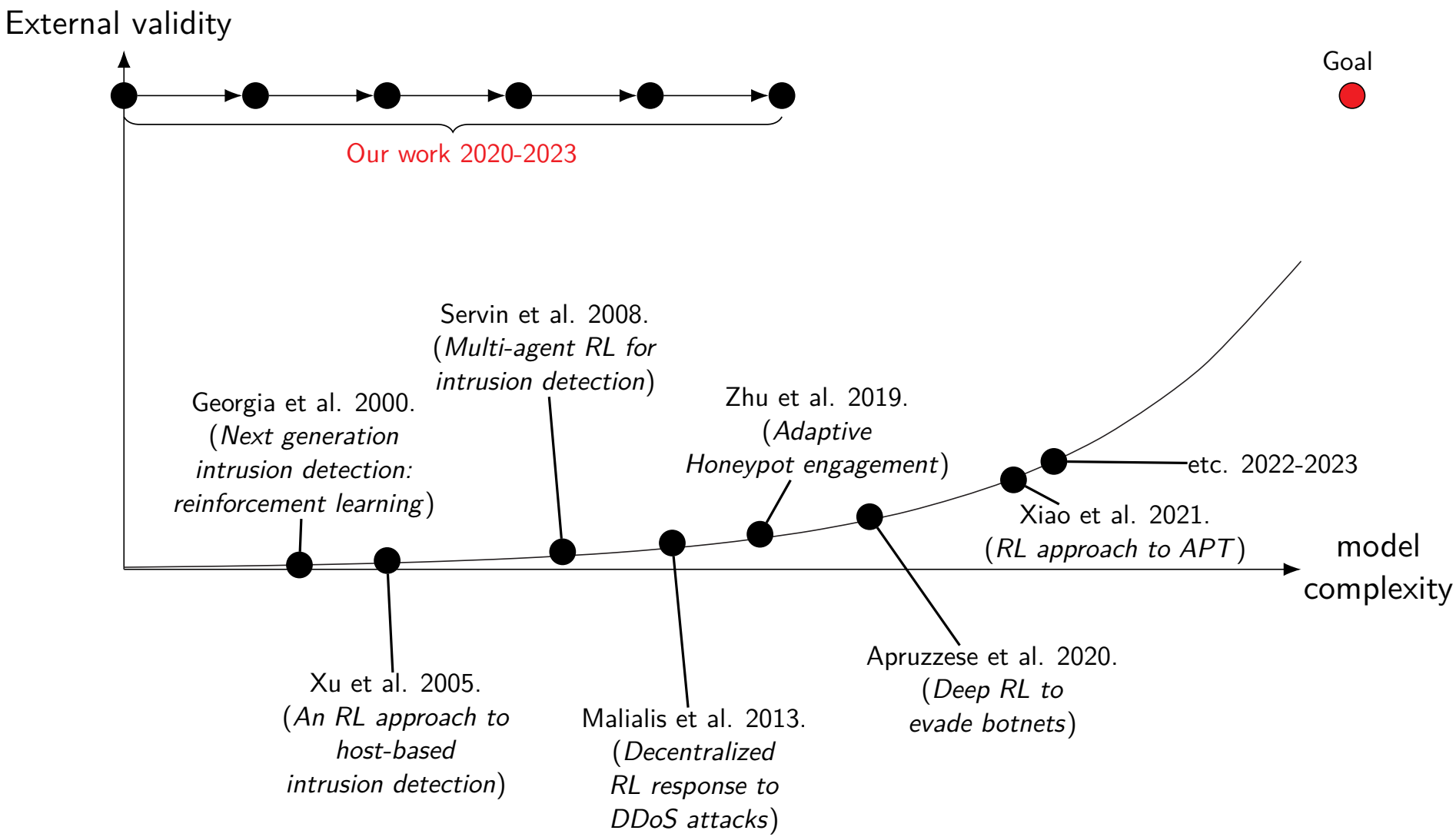
Dynamic attacker
Large set of responses



**Model complexity**

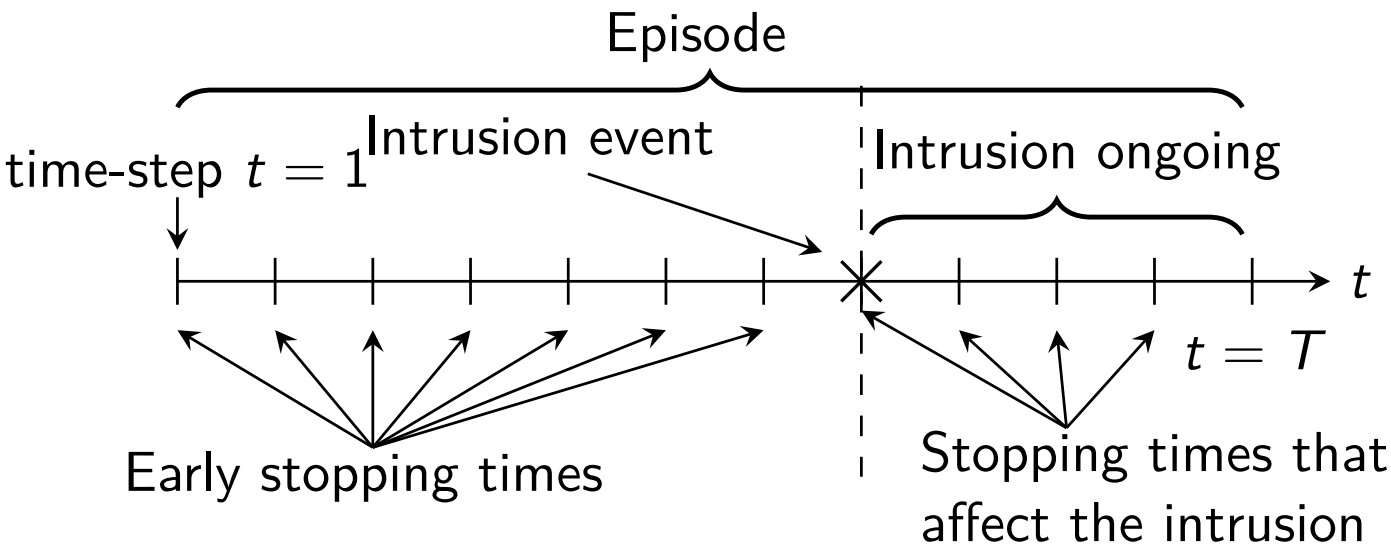
- Intrusion response can be **modeled in many ways**
  - As a *parametric optimization problem*
  - As an *optimal stopping problem*
  - As a *dynamic program*
  - As a *game*
  - etc.

# Related Work on Learning Automated Intrusion Response

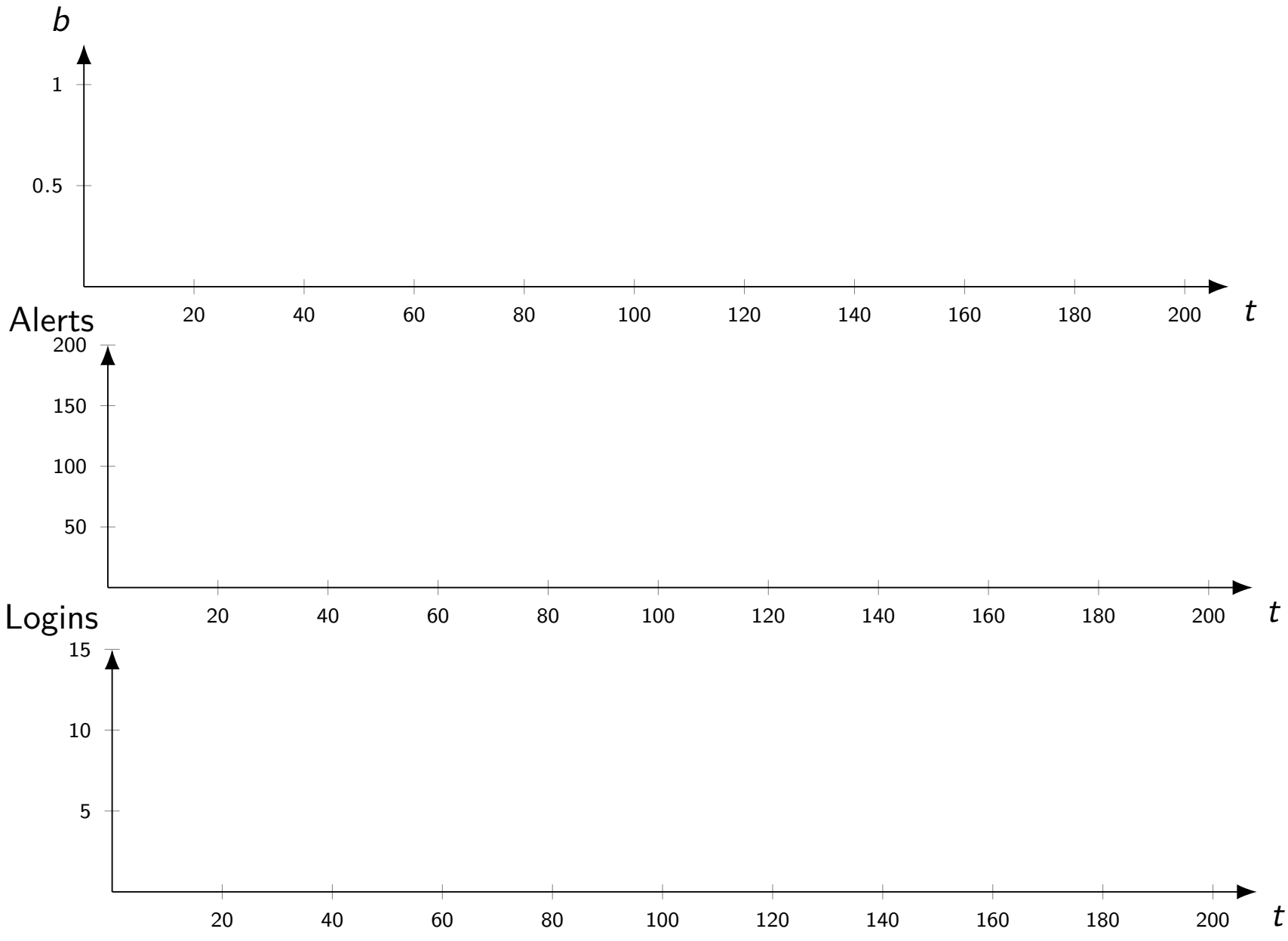


# Intrusion Response through Optimal Stopping

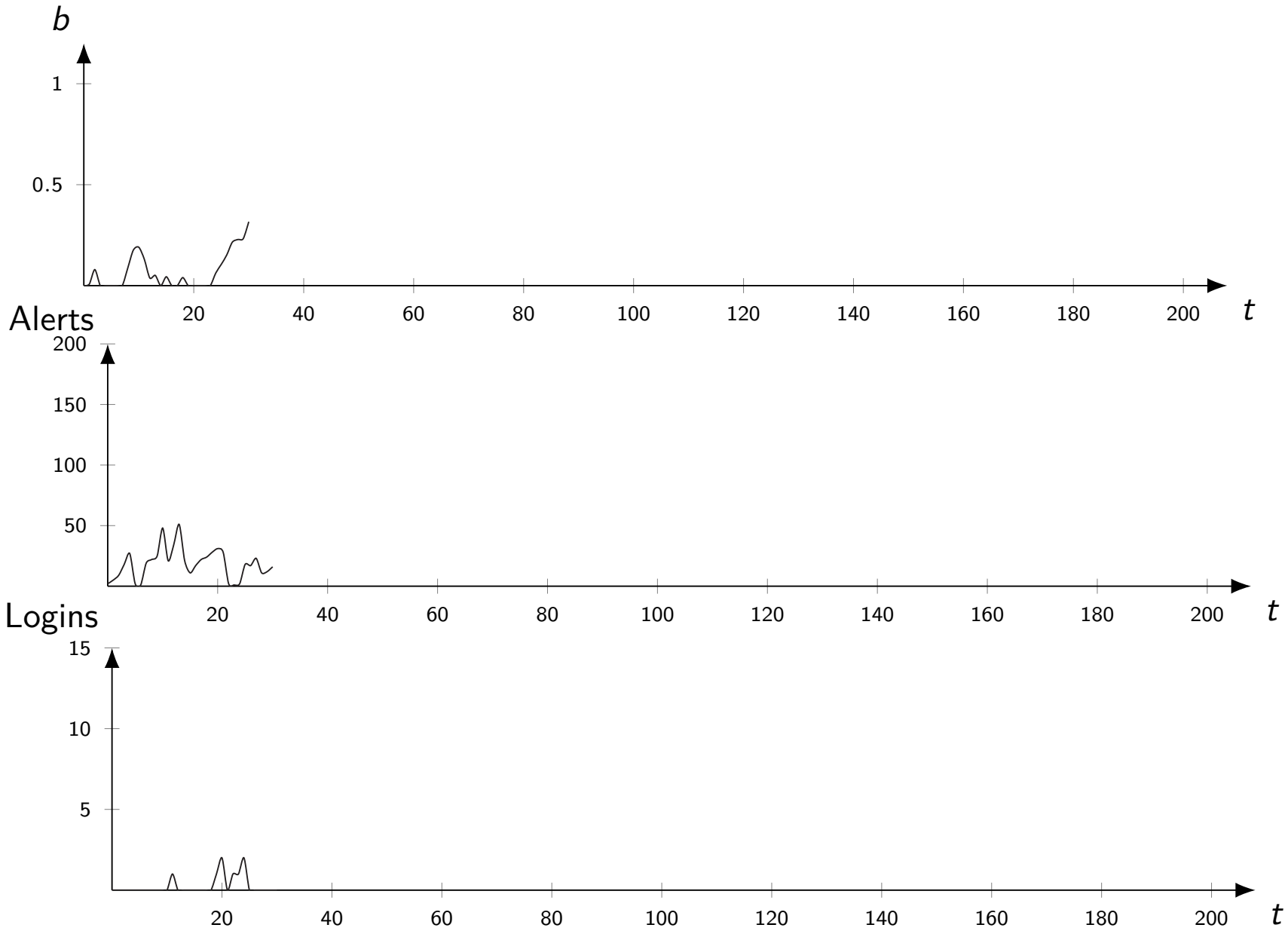
- **Suppose**
  - The attacker follows a fixed strategy (no adaptation)
  - We only have one response action, e.g., block the gateway
- Formulate intrusion response as **optimal stopping**



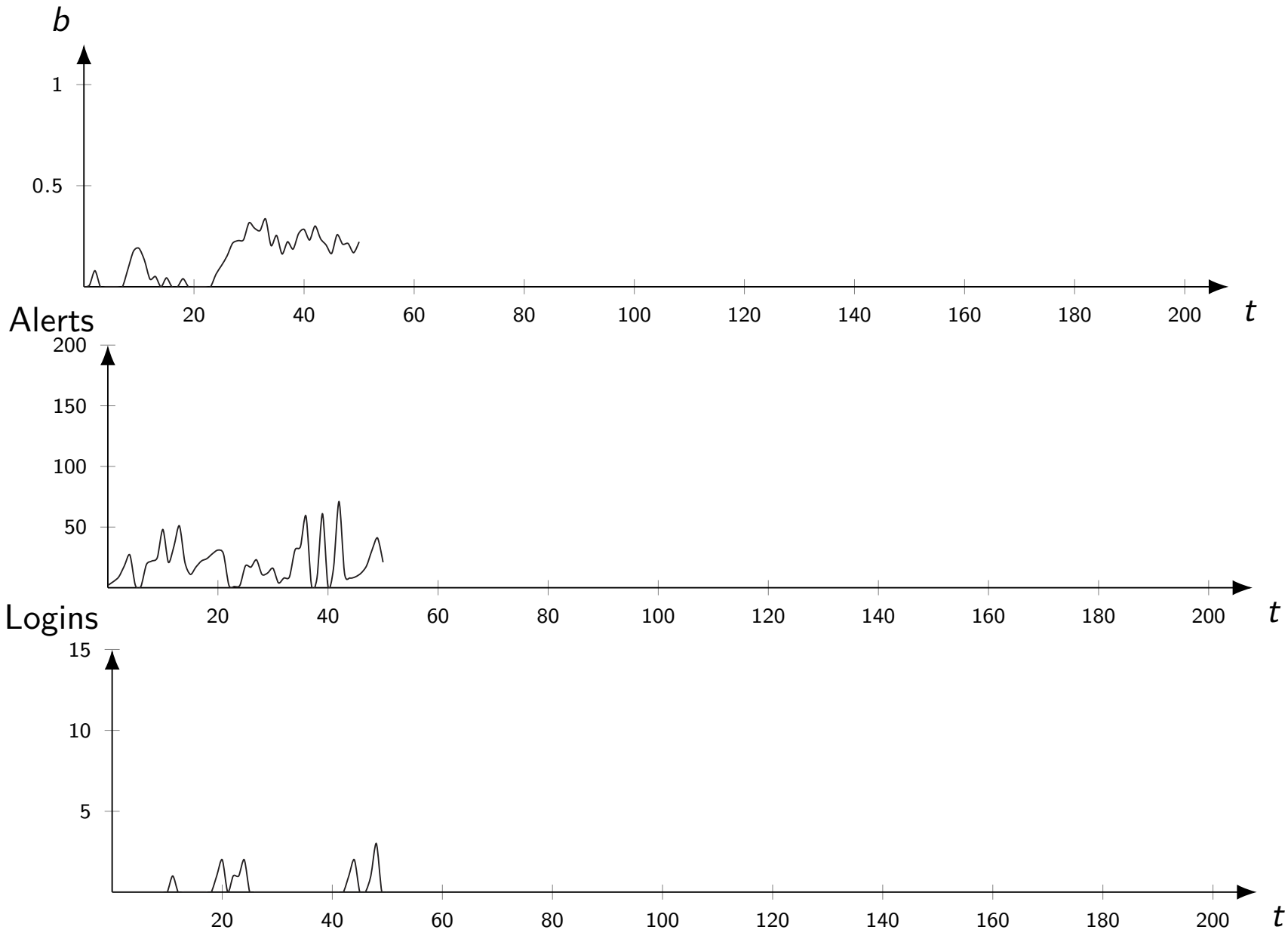
# Intrusion Response from the Defender's Perspective



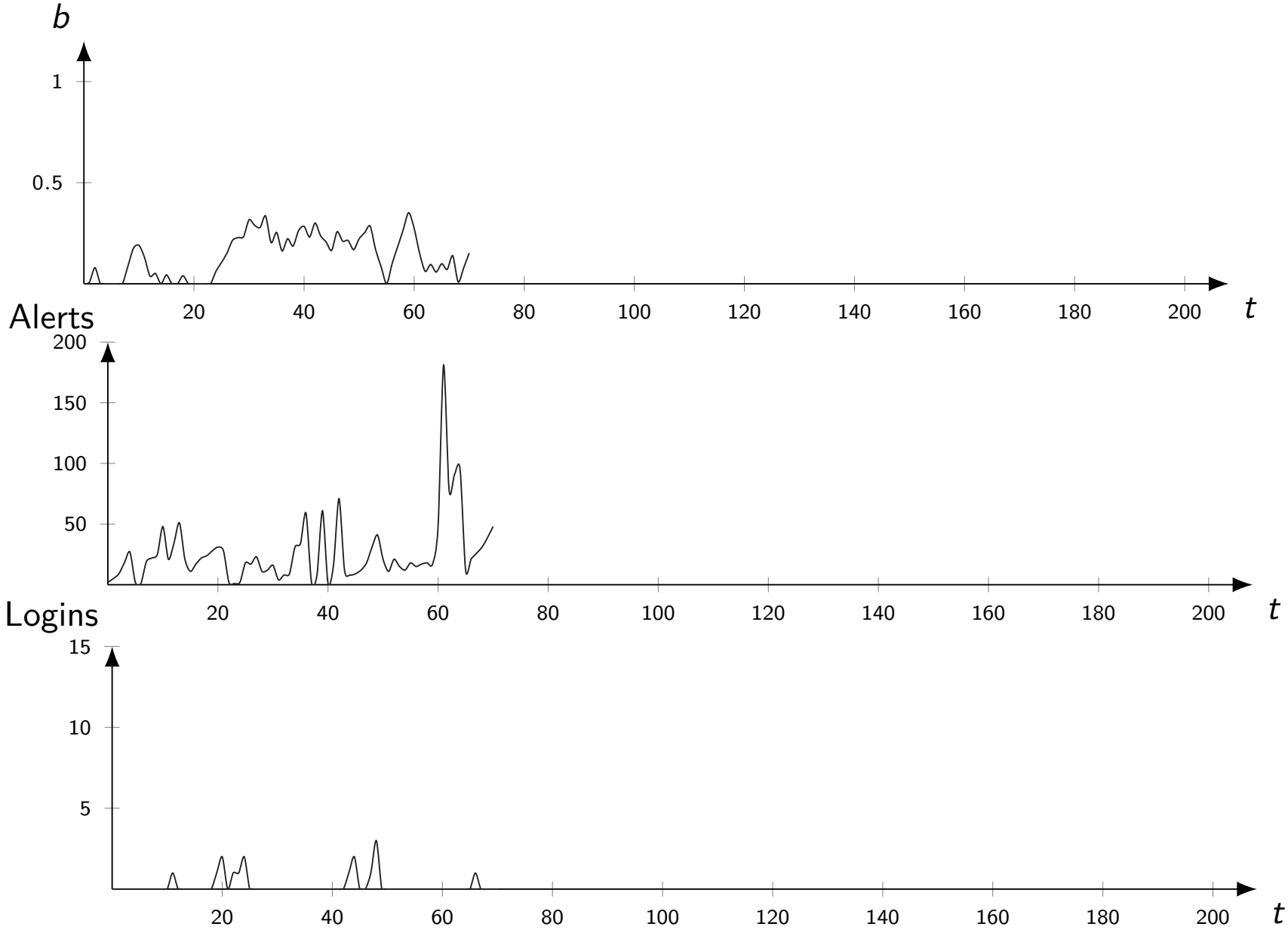
# Intrusion Response from the Defender's Perspective



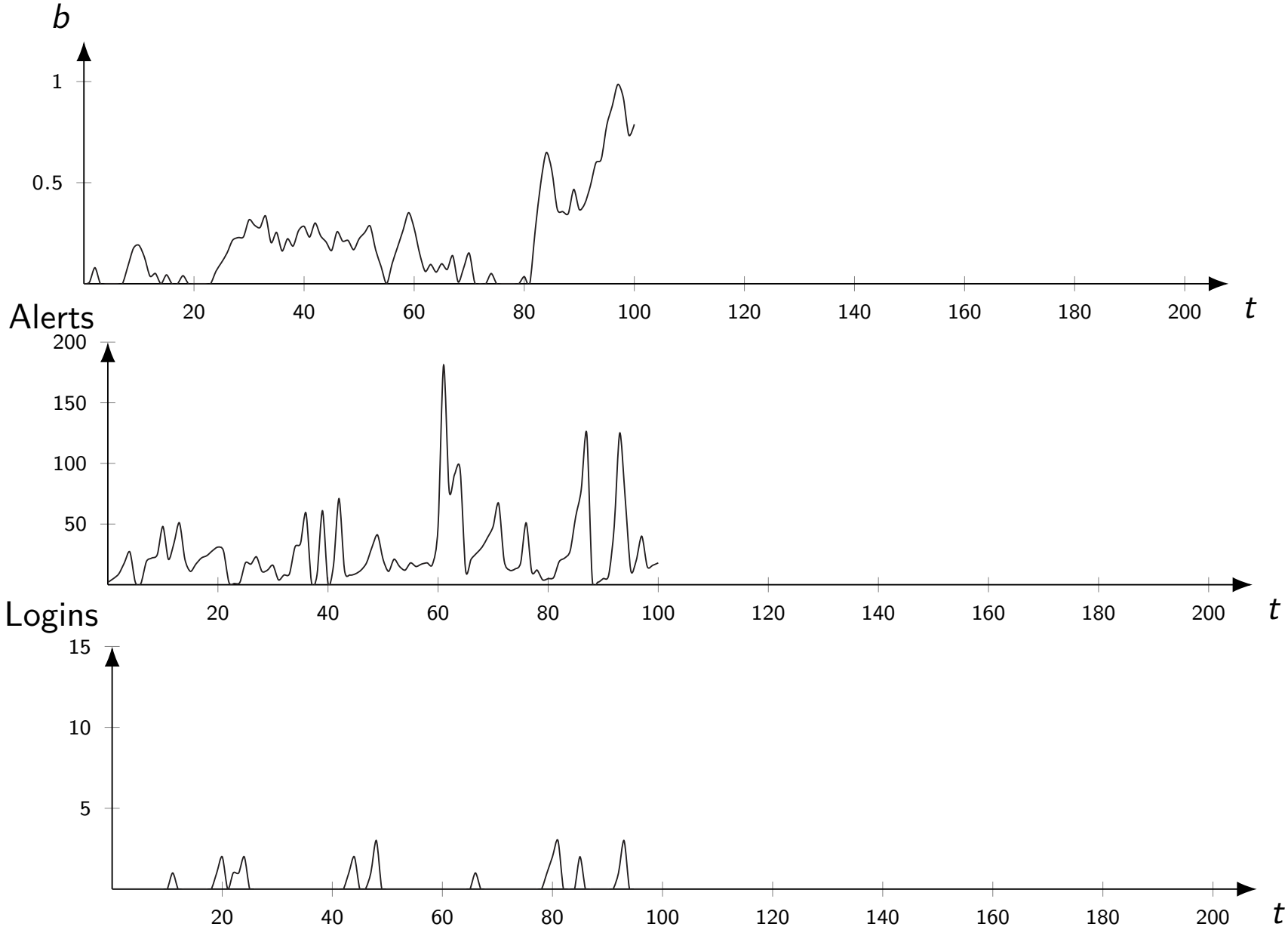
# Intrusion Response from the Defender's Perspective



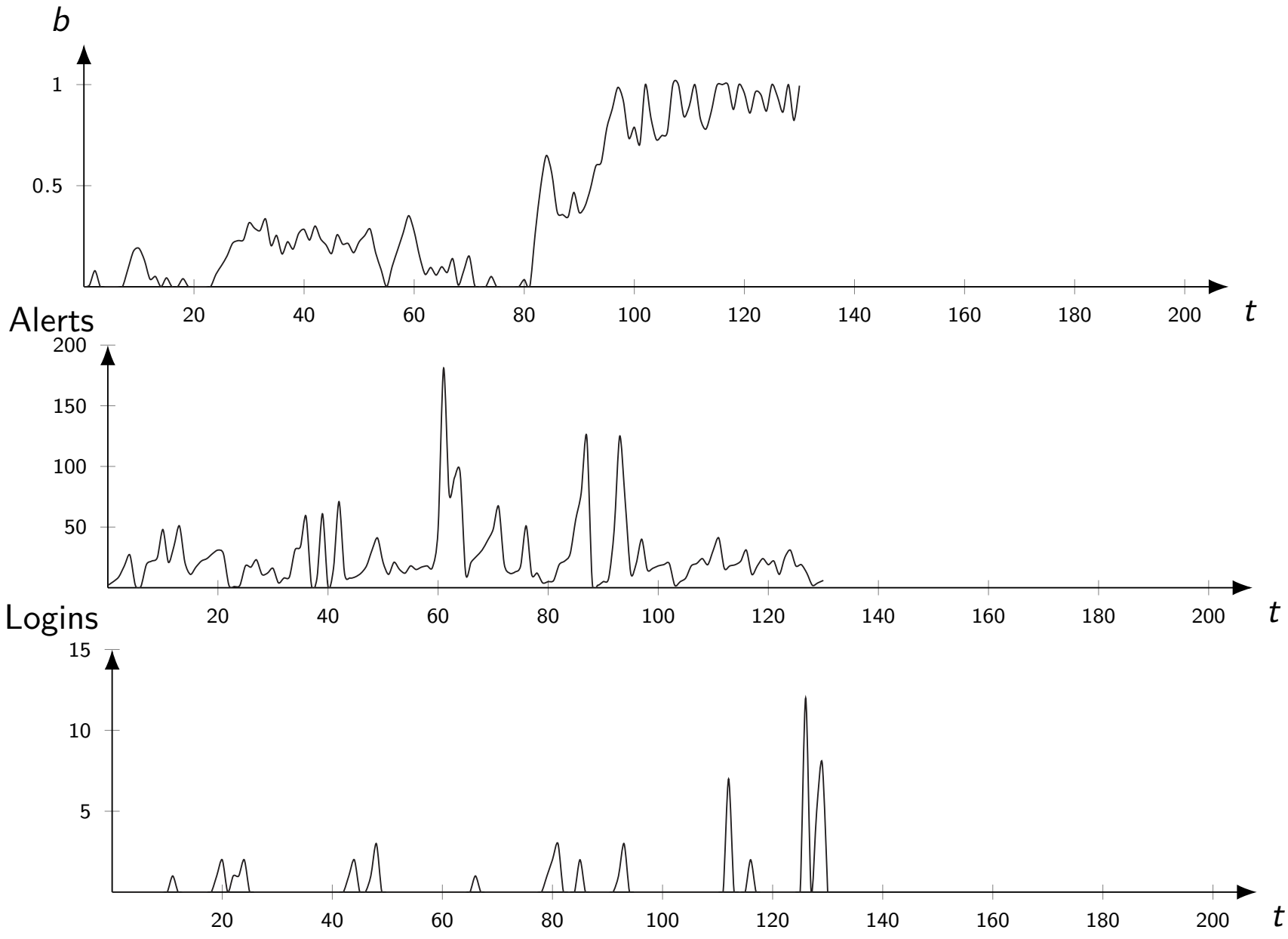
# Intrusion Response from the Defender's Perspective



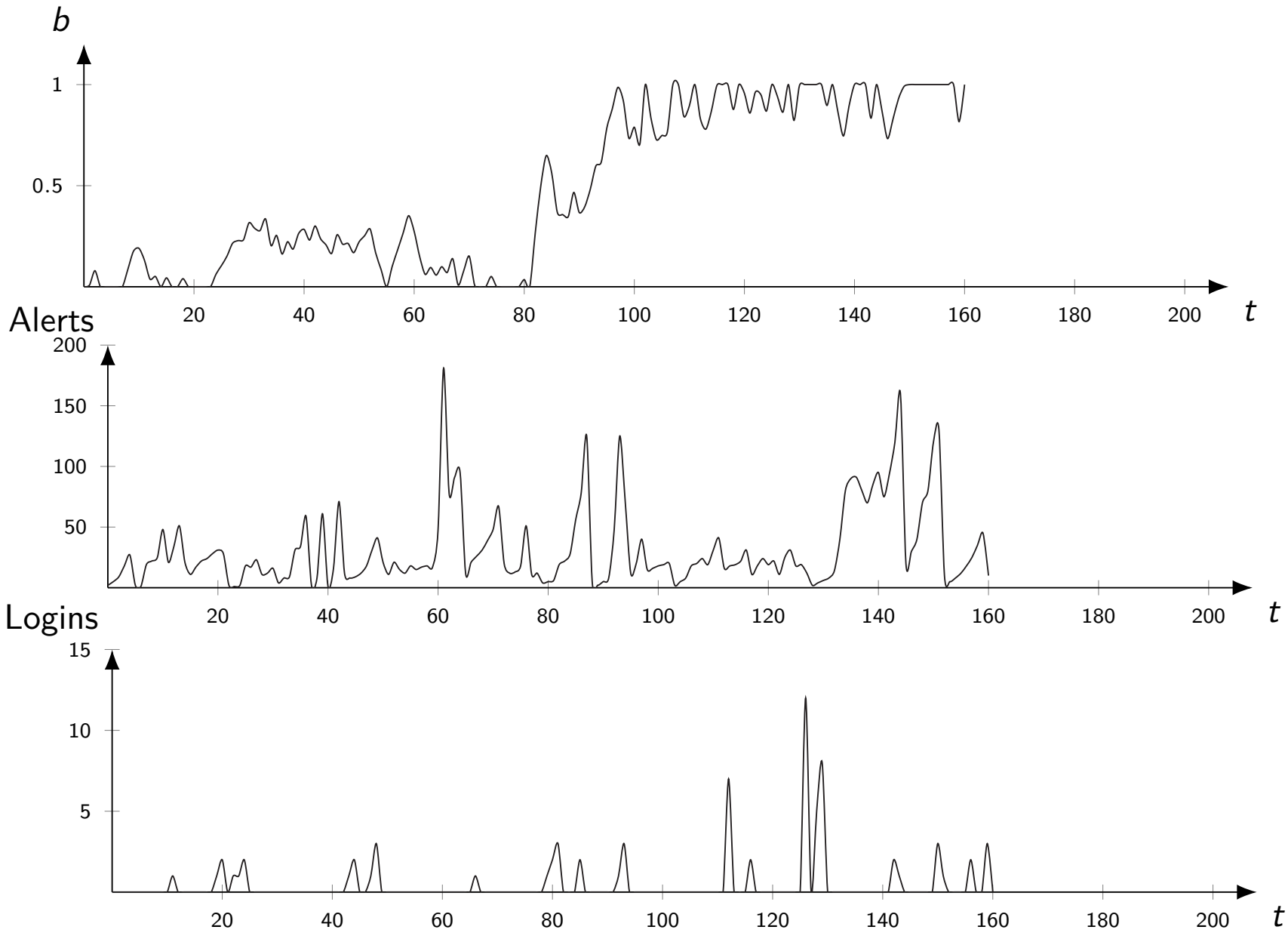
# Intrusion Response from the Defender's Perspective



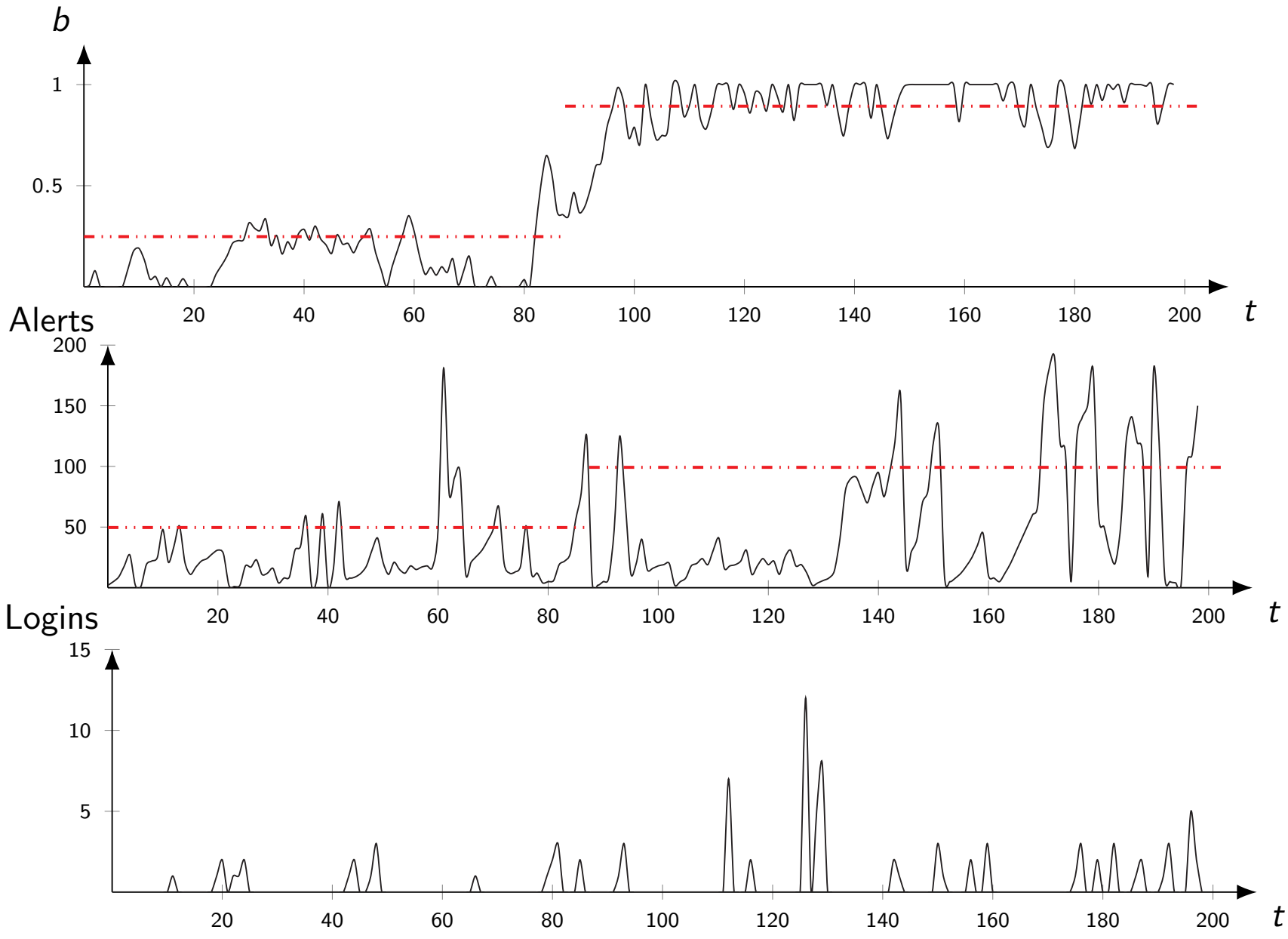
# Intrusion Response from the Defender's Perspective



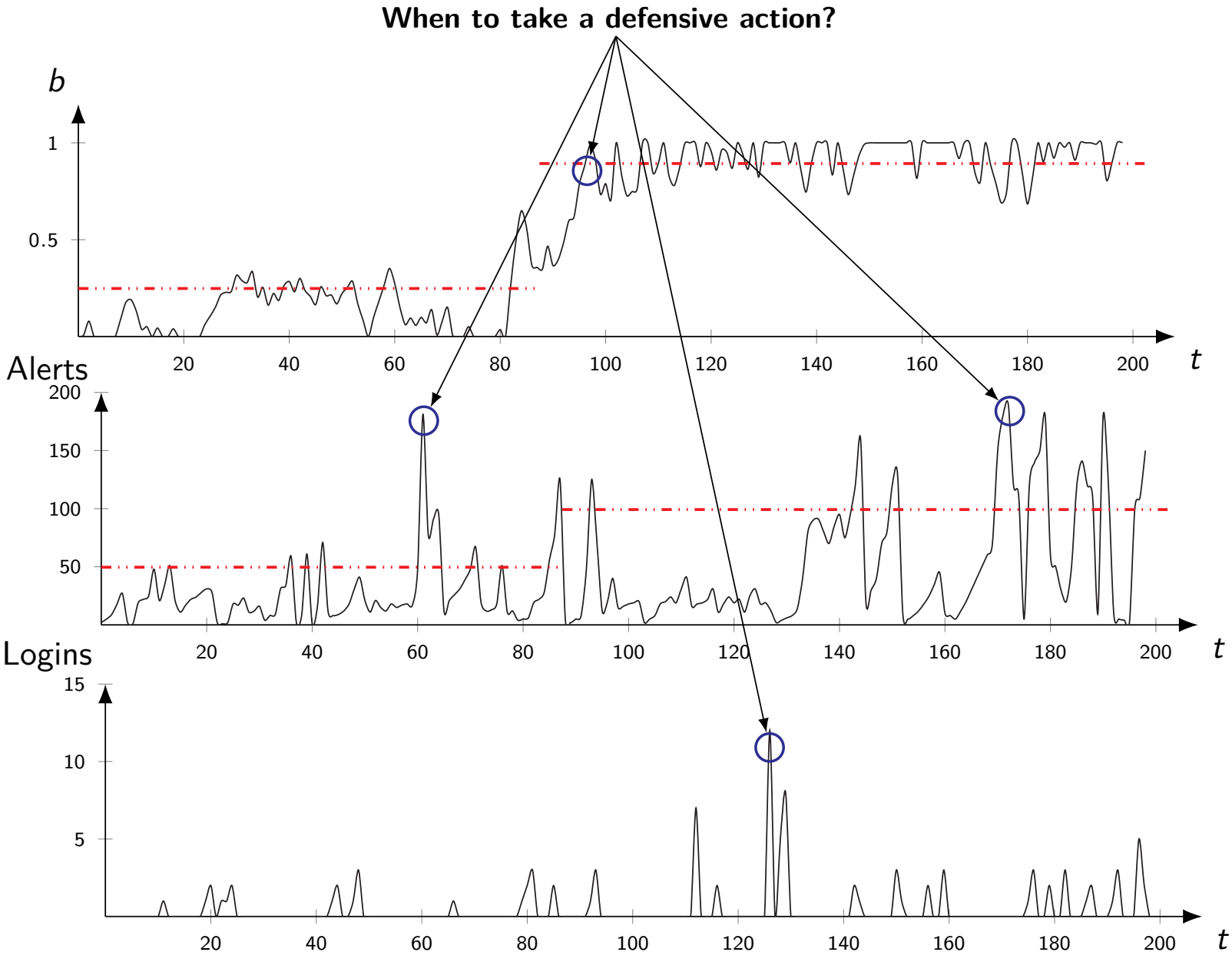
# Intrusion Response from the Defender's Perspective



# Intrusion Response from the Defender's Perspective



# Intrusion Response from the Defender's Perspective



# The Defender's Optimal Stopping Problem (1/3)

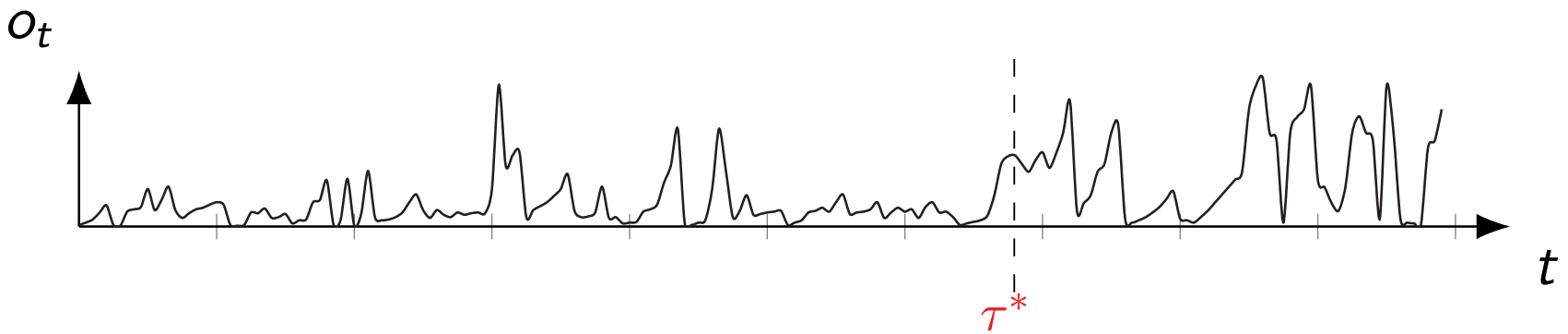
- Infrastructure is a **discrete-time dynamical system** $(s_t)_{t=1}^{T}$
- Defender observes a **noisy observation process** $(o_t)_{t=1}^{T}$
- Two options at each time $t$: $(\mathfrak{C})$ontinue and $(\mathfrak{S})$top

- Find the *optimal stopping time* $\tau^\star$:

$$\tau^\star \in \arg\max_{\tau} \mathbb{E}_\tau \left[ \sum_{t=1}^{\tau-1} \gamma^{t-1} \mathcal{R}^{\mathfrak{C}}_{s_t s_{t+1}} + \gamma^{\tau-1} \mathcal{R}^{\mathfrak{S}}_{s_\tau s_\tau} \right]$$

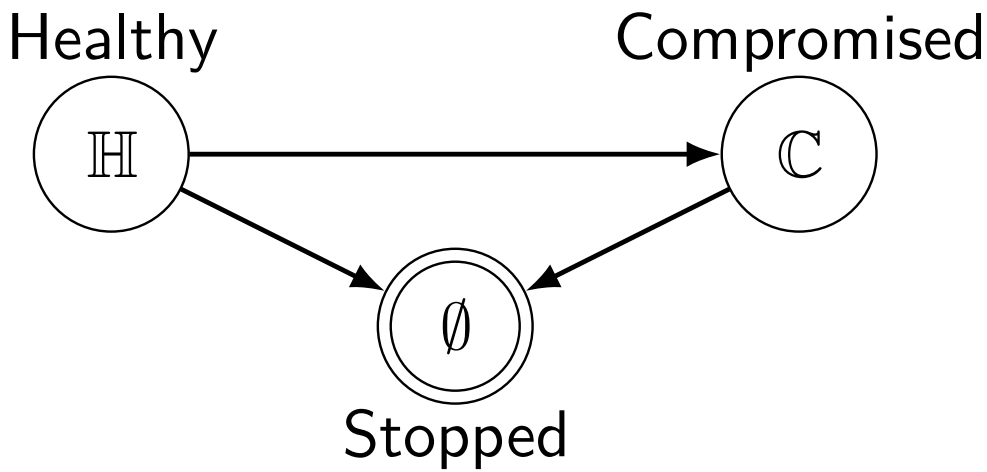
where $\mathcal{R}^{\mathfrak{S}}_{ss'}$ & $\mathcal{R}^{\mathfrak{C}}_{ss'}$ are the stop/continue rewards and $\tau$ is

$$\tau = \inf\{t : t > 0, a_t = \mathfrak{S}\}$$



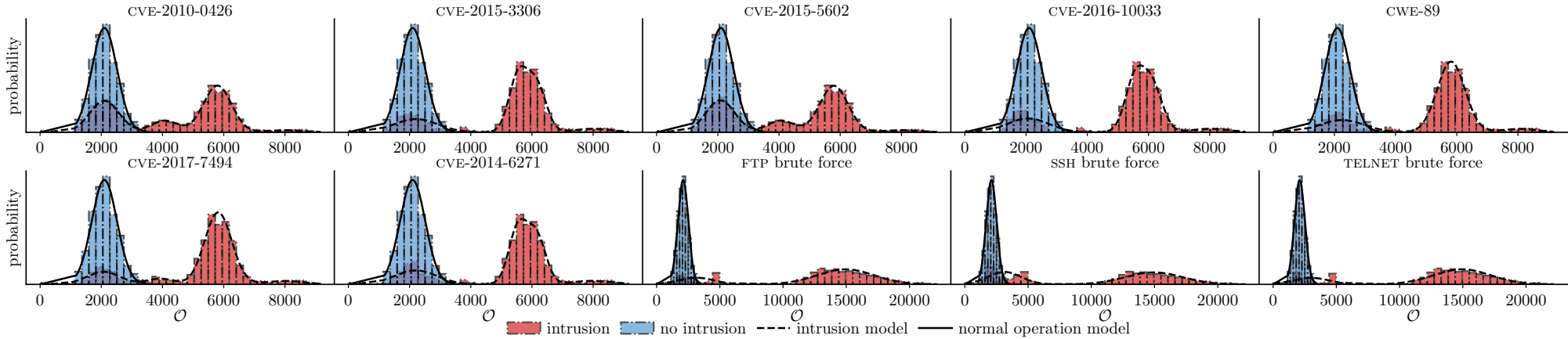
# The Defender's Optimal Stopping Problem (2/3)

- **Objective:** stop the attack as soon as possible
- Let the **state space** be $\mathcal{S} = \{\mathbb{H}, \mathbb{C}, \emptyset\}$



# The Defender's Optimal Stopping Problem (3/3)

- Let the **observation process** $(o_t)_{t=1}^{T}$ represent IDS alerts



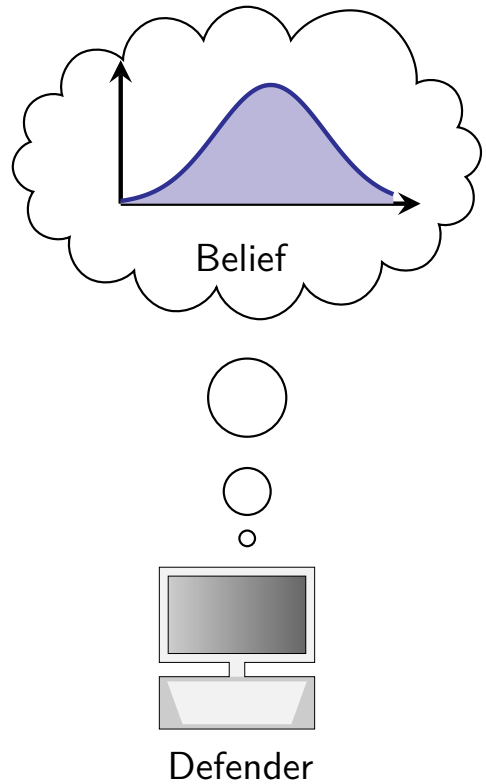

- **Estimate the observation distribution** based on $M$ samples from the twin
- E.g., compute empirical distribution $\widehat{Z}$ as estimate of $Z$
- $\widehat{Z} \rightarrow^{\text{a.s}} Z$ as $M \rightarrow \infty$ (Glivenko-Cantelli theorem)

# Optimal Stopping Strategy

- The defender can compute the **belief**

$$b_t \triangleq \mathbb{P}[S_t = \mathbb{C} \mid b_1, o_1, o_2, \ldots o_t]$$

- **Stopping strategy**:
  $\pi(b) : [0, 1] \rightarrow \{\mathfrak{S}, \mathfrak{C}\}$



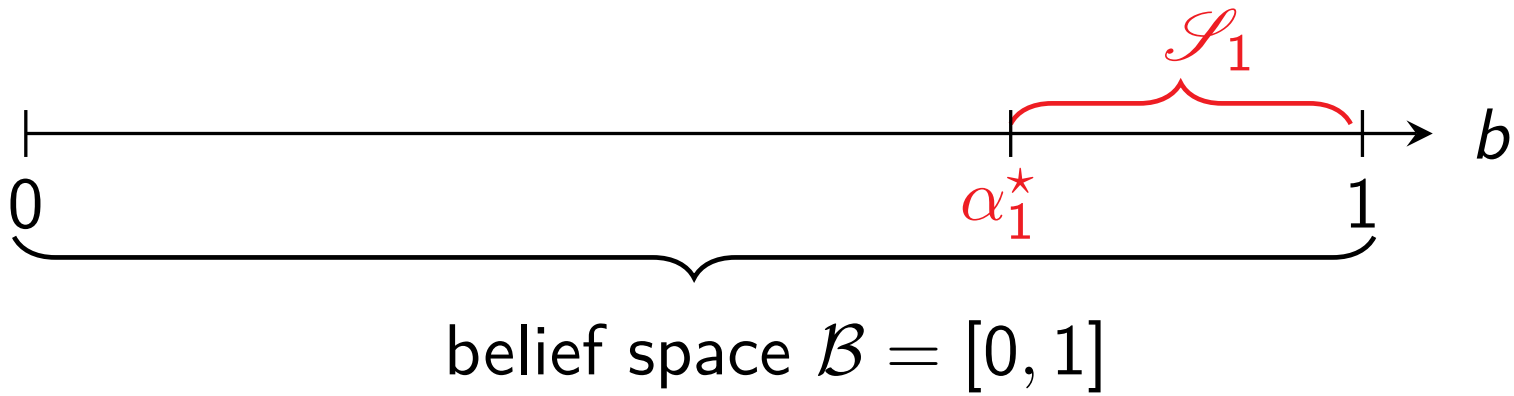
# Optimal Threshold Strategy

**Theorem**

*There exists an optimal defender strategy of the form:*

$$\pi^\star(b) = \mathfrak{S} \iff b \geq \alpha^\star \qquad\qquad \alpha^\star \in [0,1]$$

*i.e., the stopping set is* $\mathscr{S} = [\alpha^\star, 1]$

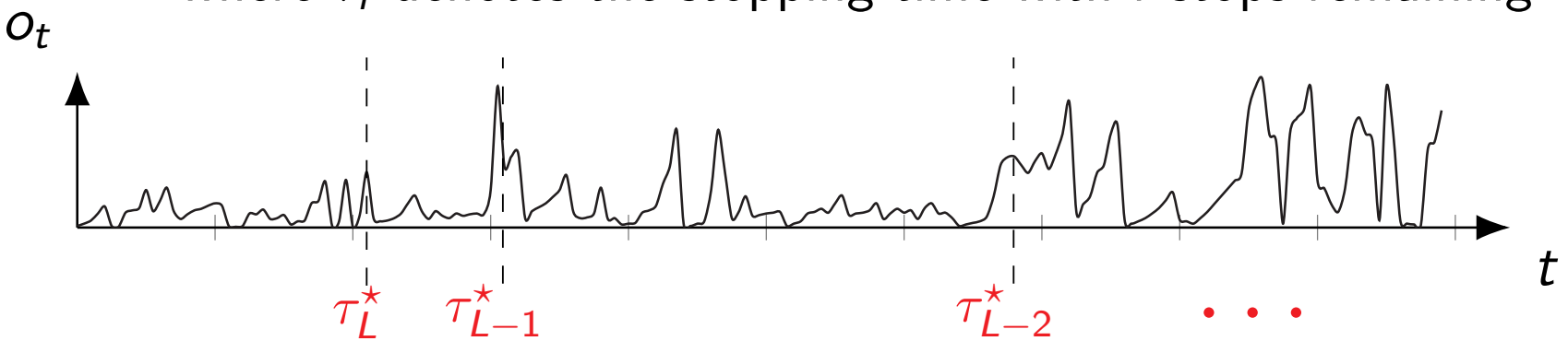


# Optimal **Multiple** Stopping

- Suppose the defender can take $L \geq 1$ **response actions**
- Find the *optimal stopping times* $\tau_L^\star, \tau_{L-1}^\star, \ldots, \tau_1^\star$:

$$(\tau_l^\star)_{l=1,\ldots,L} \in \underset{\tau_1,\ldots,\tau_L}{\arg\max}\, \mathbb{E}_{\tau_1,\ldots,\tau_L}\Bigg[\sum_{t=1}^{\tau_L-1} \gamma^{t-1}\mathcal{R}^{\mathfrak{C}}_{s_t s_{t+1}} + \gamma^{\tau_L-1}\mathcal{R}^{\mathfrak{S}}_{s_{\tau_L} s_{\tau_L}} + \sum_{t=\tau_L+1}^{\tau_{L-1}-1} \gamma^{t-1}\mathcal{R}^{\mathfrak{C}}_{s_t s_{t+1}} + \gamma^{\tau_{L-1}-1}\mathcal{R}^{\mathfrak{S}}_{s_{\tau_{L-1}} s_{\tau_{L-2}}} + \ldots + \sum_{t=\tau_2+1}^{\tau_1-1} \gamma^{t-1}\mathcal{R}^{\mathfrak{C}}_{s_t s_{t+1}} + \gamma^{\tau_1-1}\mathcal{R}^{\mathfrak{S}}_{s_{\tau_1} s_{\tau_1}}\Bigg]$$

where $\tau_l$ denotes the stopping time with $l$ stops remaining.



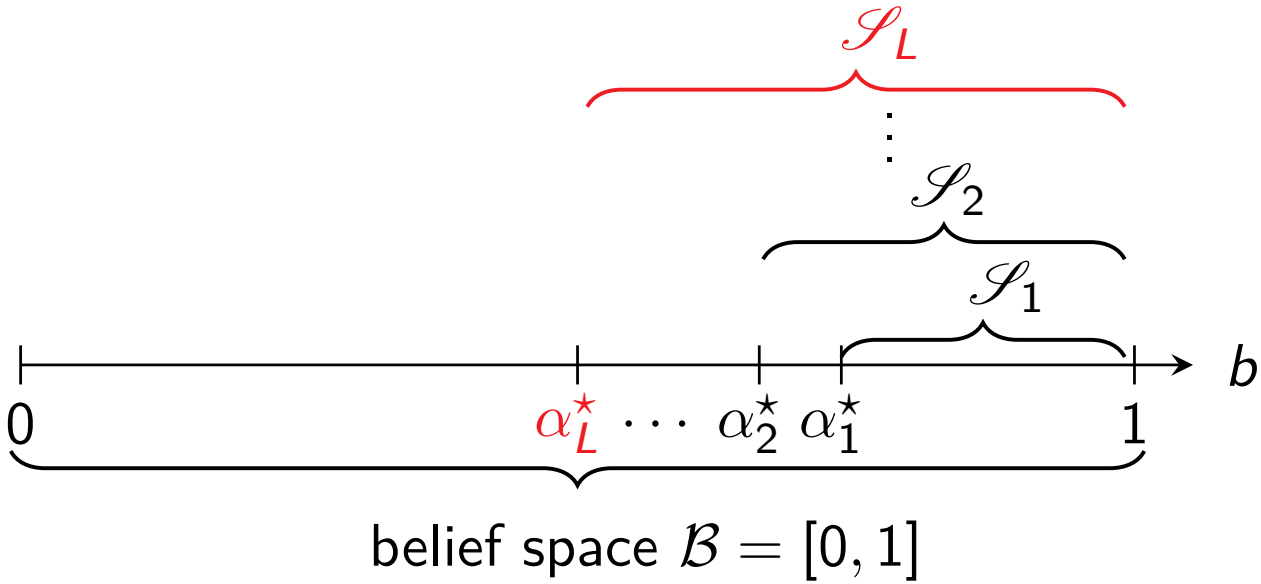
# Optimal **Multi**-Threshold Strategy

## Theorem

- *Stopping sets are nested* $\mathscr{S}_{l-1} \subseteq \mathscr{S}_l$ *for* $l = 2, \dots L$.
- *If* $(o_t)_{t\geq 1}$ *is totally positive of order 2 (TP2), there exists an optimal defender strategy of the form:*

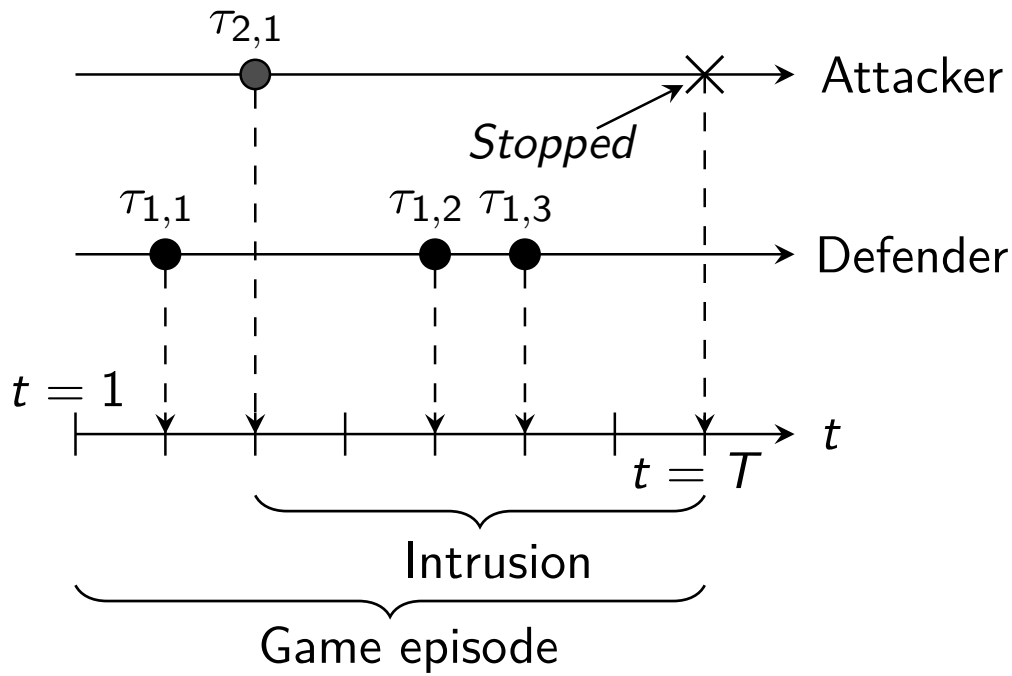
$$\pi_l^\star(b) = \mathfrak{S} \iff b \geq \alpha_l^\star, \qquad l = 1, \dots, L$$

*where* $\alpha_l^\star \in [0,1]$ *is decreasing in* $l$.



# Optimal Stopping **Game**

- Suppose the attacker is dynamic and **decides when to start and abort** its intrusion.




- Find the *optimal stopping times*

$$\underset{\tau_{\mathrm{D},1},\ldots,\tau_{\mathrm{D},L}}{\text{maximize}} \; \underset{\tau_{\mathrm{A},1},\tau_{\mathrm{A},2}}{\text{minimize}} \; \mathbb{E}[J]$$

where $J$ is the defender's objective.

# Best-Response Multi-Threshold Strategies (1/2)

**Theorem**

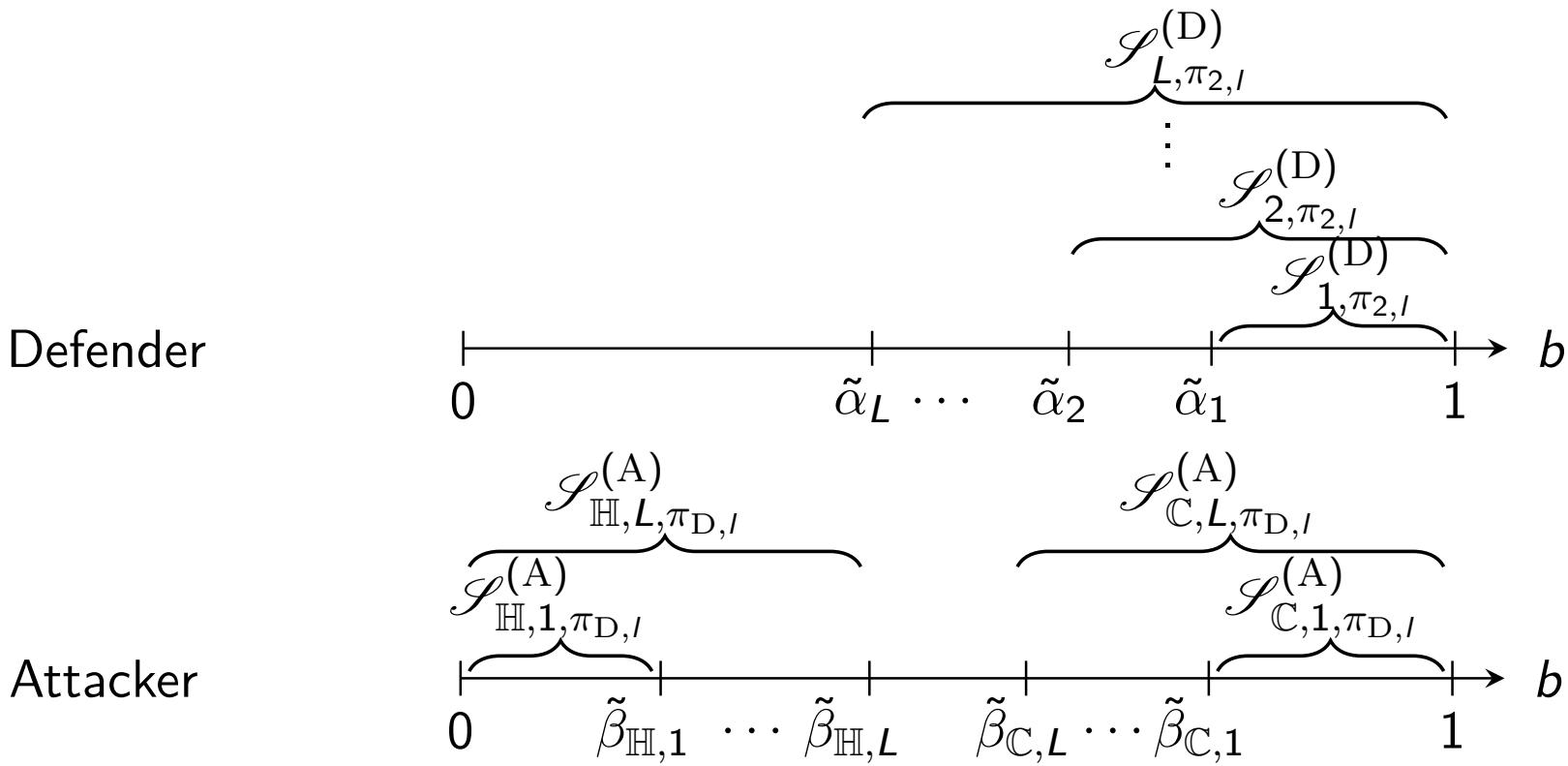
- *The defender's best response is of the form:*

$$\tilde{\pi}_{\mathrm{D},l}(b) = \mathfrak{S} \iff b \geq \tilde{\alpha}_l, \qquad l = 1, \ldots, L$$

- *The attacker's best response is of the form:*

$$\tilde{\pi}_{\mathrm{A},l}(b) = \mathfrak{C} \iff \tilde{\pi}_{\mathrm{D},l}(\mathfrak{S} \mid b) \geq \tilde{\beta}_{\mathbb{H},l}, \quad l = 1, \ldots, L, s = \mathbb{H}$$
$$\tilde{\pi}_{\mathrm{A},l}(b) = \mathfrak{S} \iff \tilde{\pi}_{\mathrm{D},l}(\mathfrak{S} \mid b) \geq \tilde{\beta}_{\mathbb{C},l}, \quad l = 1, \ldots, L, s = \mathbb{C}$$

# Best-Response Multi-Threshold Strategies (2/2)



# Efficient Computation of Best Responses

**Algorithm 1:** Threshold Optimization

**Input:** Objective function $J$, number of thresholds $L$, parametric optimizer $\mathrm{PO}$

**Output:** A approximate best response strategy $\hat{\pi}_\theta$

**Algorithm**

$\Theta \leftarrow [0,1]^L$

For each $\theta \in \Theta$, define $\pi_\theta(b_t)$ as

$$\pi_\theta(b_t) \triangleq \begin{cases} \mathfrak{S} & \text{if } b_t \geq \theta_i \\ \mathfrak{C} & \text{otherwise} \end{cases}$$

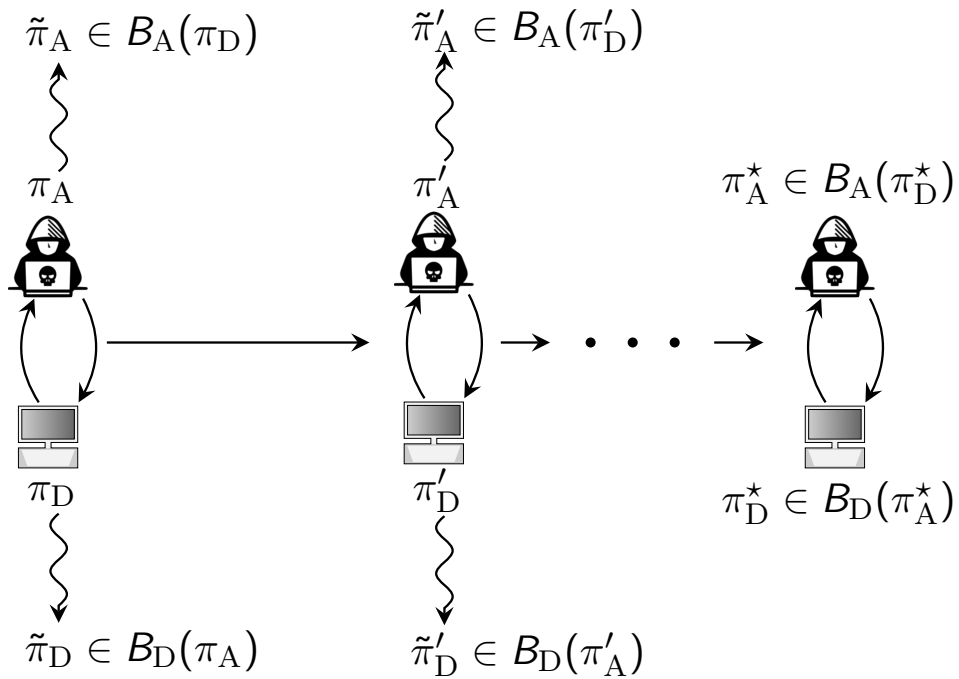
$J_\theta \leftarrow \mathbb{E}_{\pi_\theta}[J]$

$\hat{\pi}_\theta \leftarrow \mathrm{PO}(\Theta, J_\theta)$

**return** $\hat{\pi}_\theta$

- Examples of parameteric optimization algorithmns: CEM, BO, CMA-ES, DE, SPSA, etc.

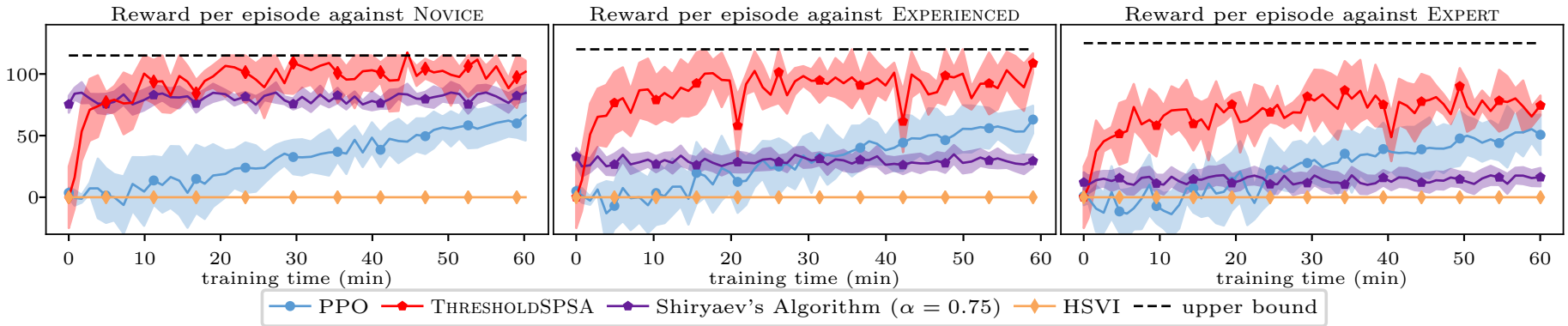
# Threshold-Fictitious Play to Approximate an Equilibrium

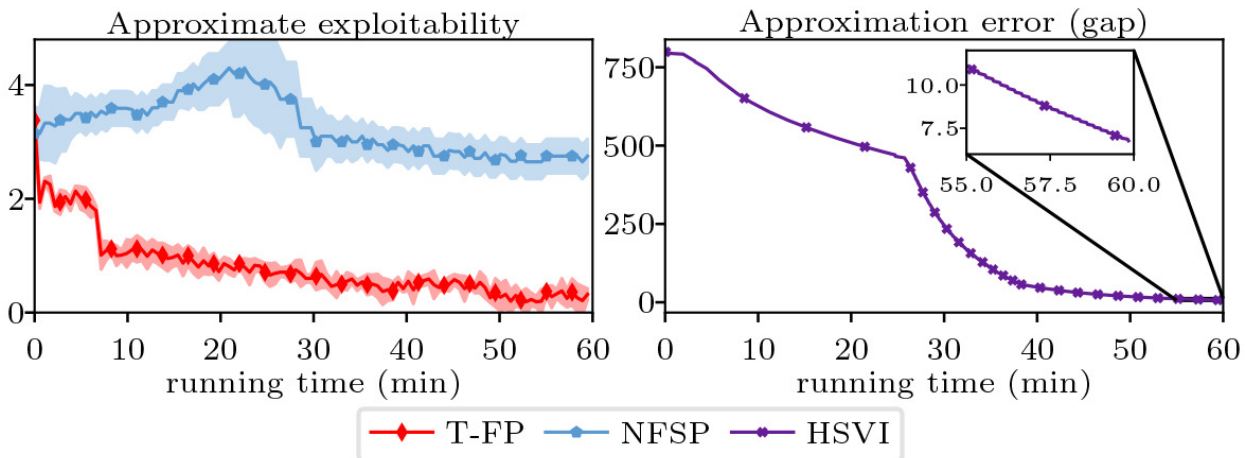



Fictitious play: iterative averaging of best responses.

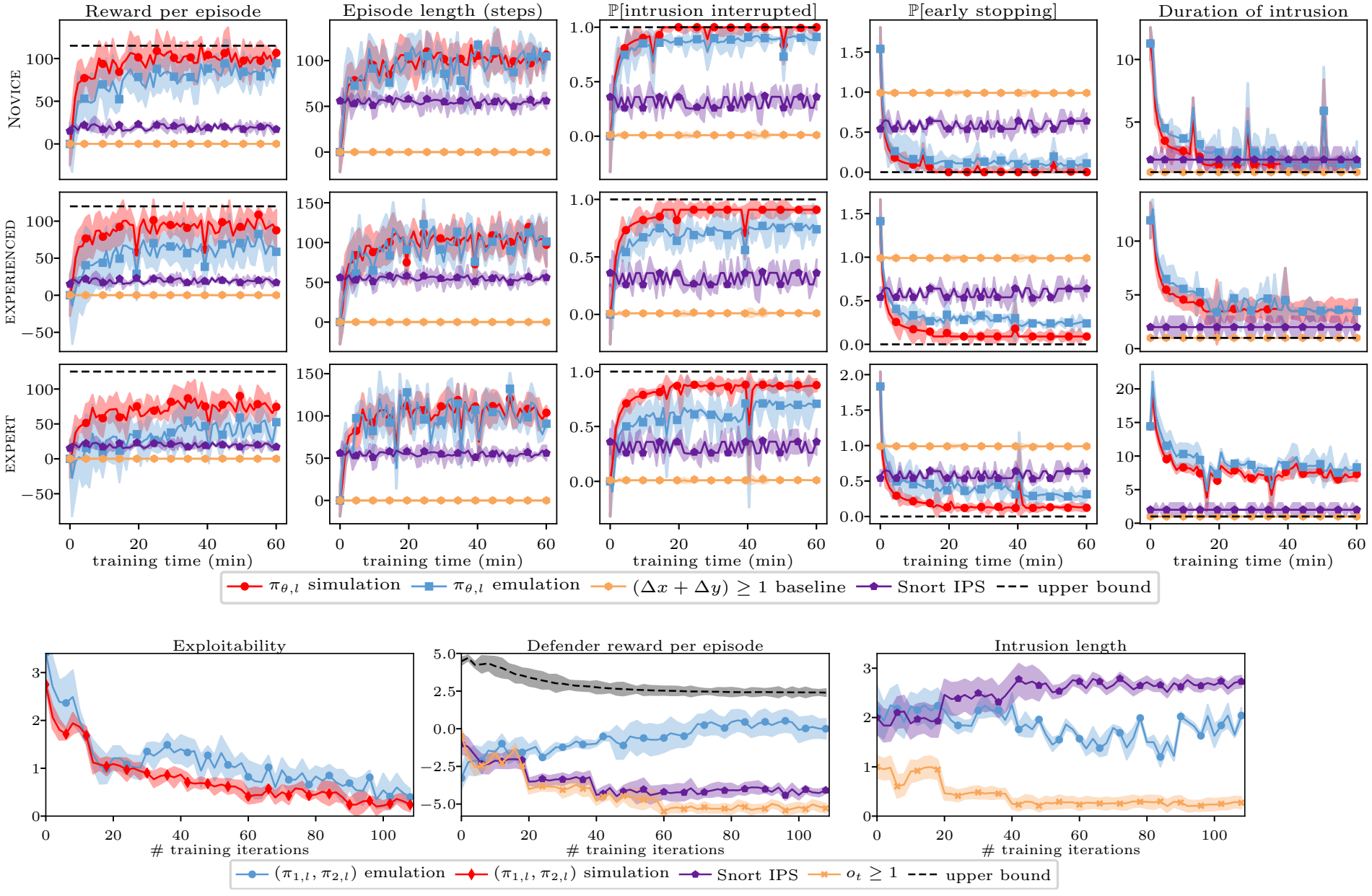
- **Learn best response** strategies iteratively
- Average best responses to **approximate the equilibrium**

# Comparison against State-of-the-art Algorithms

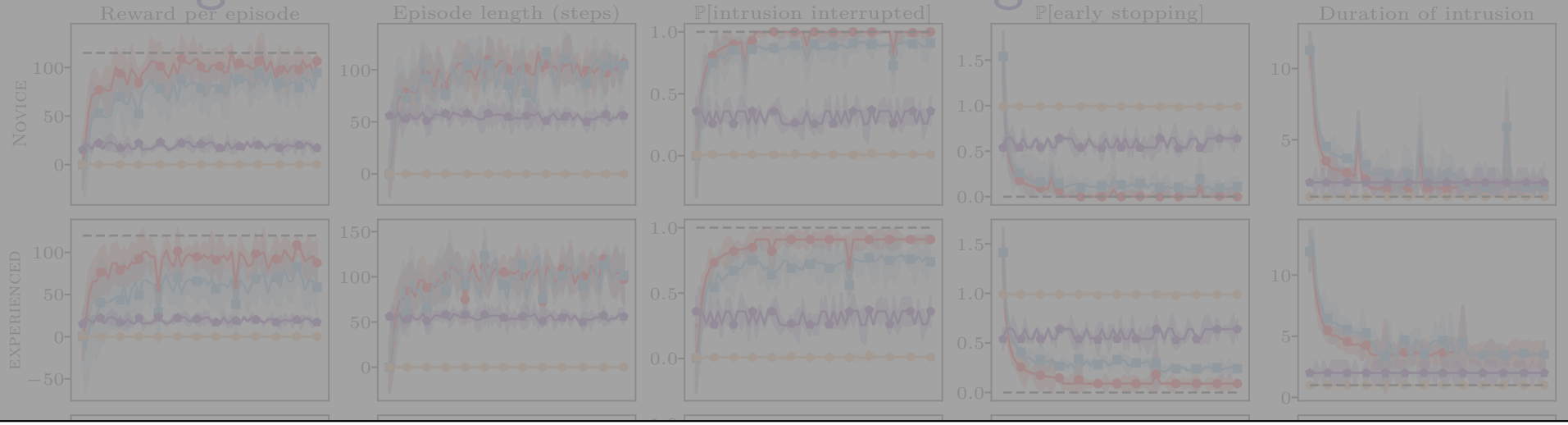


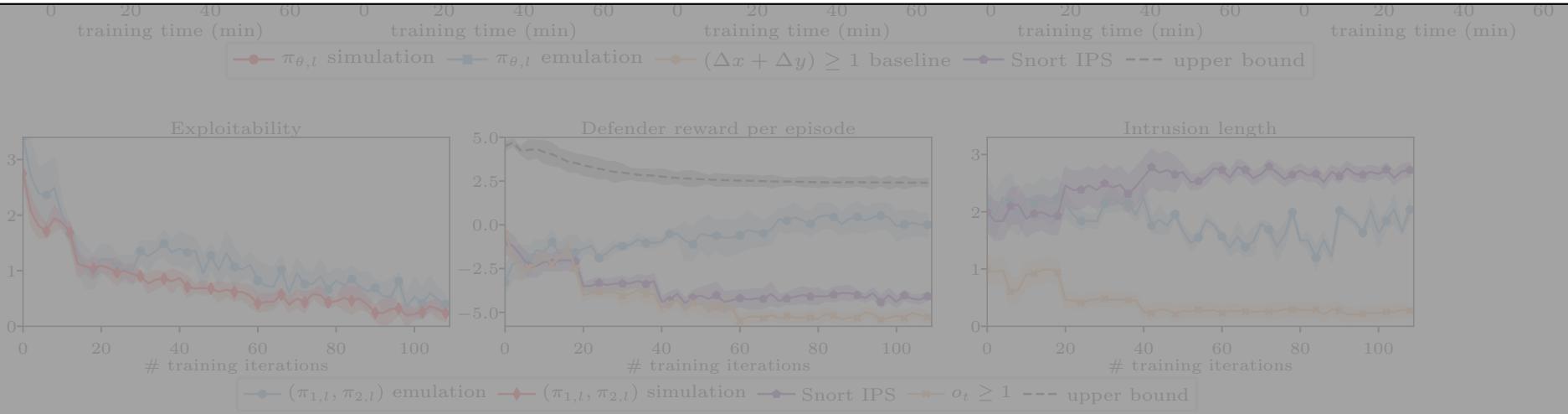
# Learning Curves in Simulation and Digital Twin



# Learning Curves in Simulation and Digital Twin

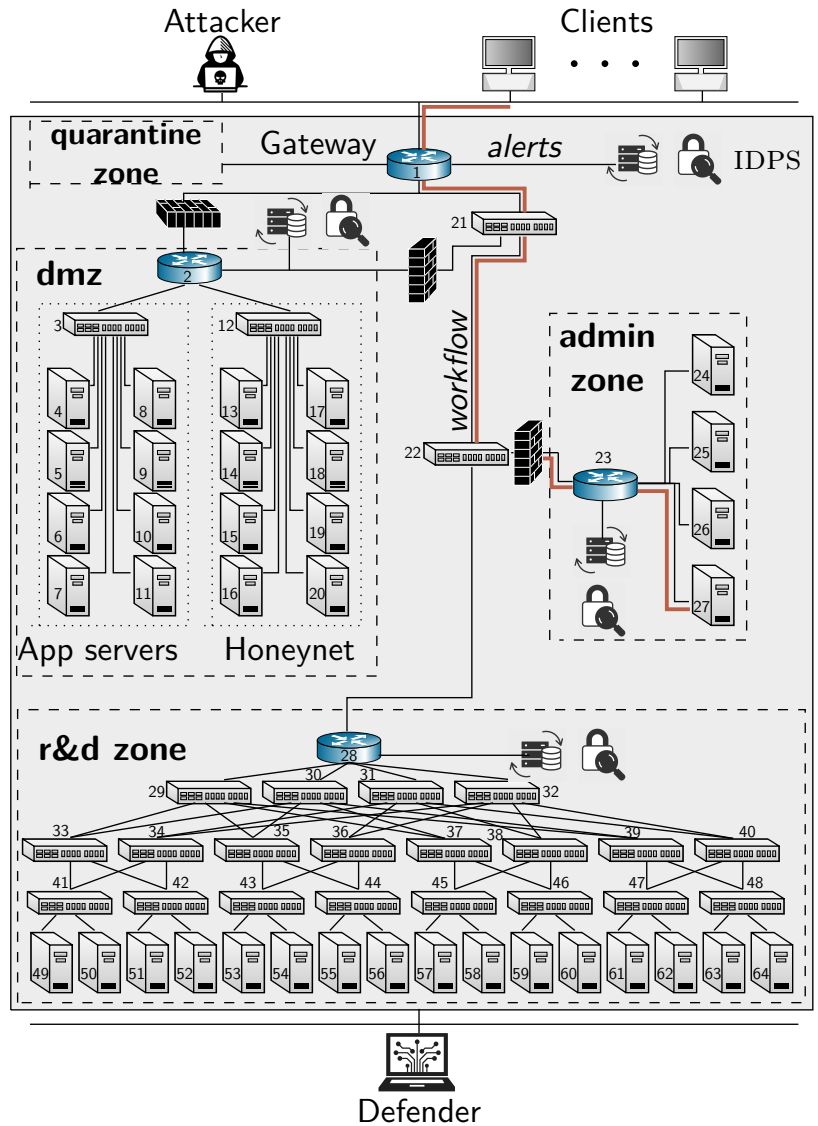



Stopping is about **timing**; now we consider timing + action selection



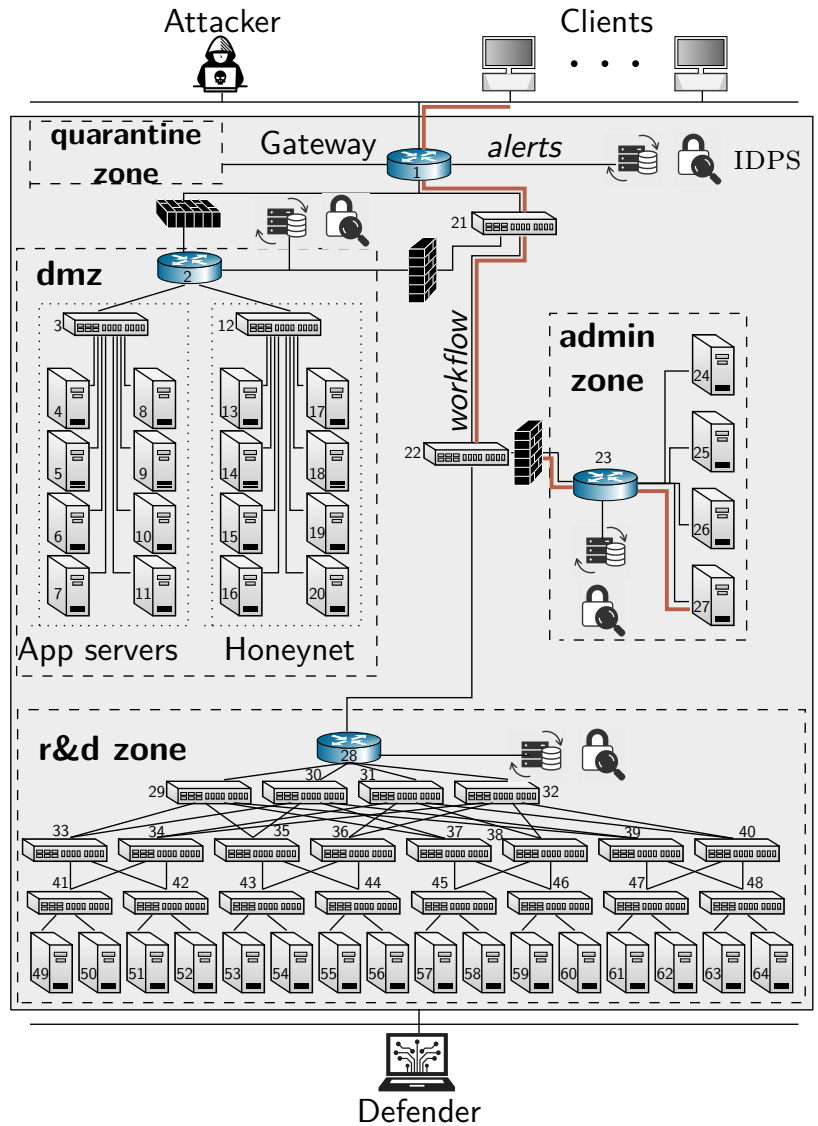
# **General** Intrusion Response Game

- Suppose the defender and the attacker can take $L$ actions **per node**
- $\mathcal{G} = \langle \{\text{gw}\} \cup \mathcal{V}, \mathcal{E} \rangle$: directed tree representing the virtual infrastructure
- $\mathcal{V}$: set of virtual nodes
- $\mathcal{E}$: set of node dependencies
- $\mathcal{Z}$: set of zones



# **General** Intrusion Response Game

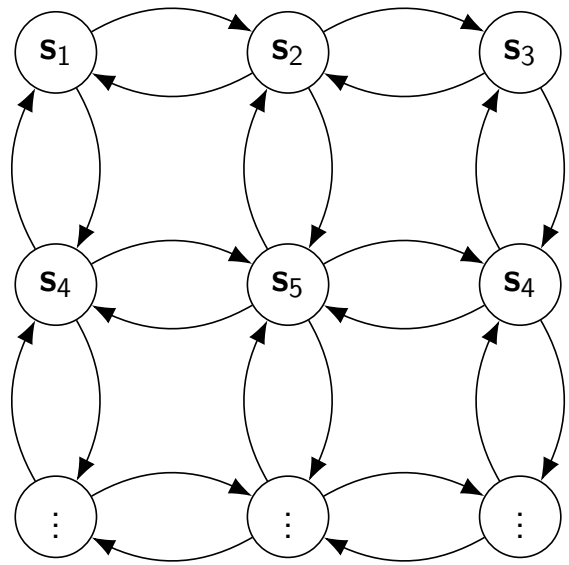
- Suppose the defender and the attacker can take $L$ actions **per node**
- $\mathcal{G} = \langle \{\text{gw}\} \cup \mathcal{V}, \mathcal{E} \rangle$: directed tree representing the virtual infrastructure
- $\mathcal{V}$: set of virtual nodes
- $\mathcal{E}$: set of node dependencies
- $\mathcal{Z}$: set of zones



# State Space

- Each $i \in \mathcal{V}$ has a state

$$\boldsymbol{v}_{i,t} = (\underbrace{v_{t,i}^{(\mathrm{Z})}}_{\mathrm{D}}, \underbrace{v_{t,i}^{(\mathrm{I})}, v_{t,i}^{(\mathrm{R})}}_{\mathrm{A}})$$




- System state $\mathbf{s}_t = (\mathbf{v}_{t,i})_{i \in \mathcal{V}} \sim \mathbf{S}_t$
- Markovian time-homogeneous dynamics:

$$\mathbf{s}_{t+1} \sim f(\cdot \mid \mathbf{S}_t, \mathbf{A}_t)$$

$\mathbf{A}_t = (\mathbf{A}_t^{(\mathrm{A})}, \mathbf{A}_t^{(\mathrm{D})})$ are the actions.

# State Space

- Each $i \in \mathcal{V}$ has a state

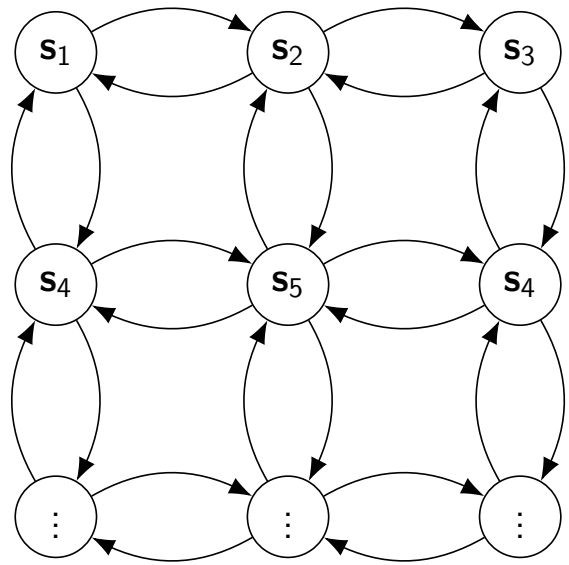
$$\boldsymbol{v}_{i,t} = (\underbrace{v_{t,i}^{(\mathrm{Z})}}_{\mathrm{D}}, \underbrace{v_{t,i}^{(\mathrm{I})}, v_{t,i}^{(\mathrm{R})}}_{\mathrm{A}})$$

- System state $\mathbf{s}_t = (\mathbf{v}_{t,i})_{i \in \mathcal{V}} \sim \mathbf{S}_t$

- Markovian time-homogeneous dynamics:

$$\mathbf{s}_{t+1} \sim f(\cdot \mid \mathbf{S}_t, \mathbf{A}_t)$$

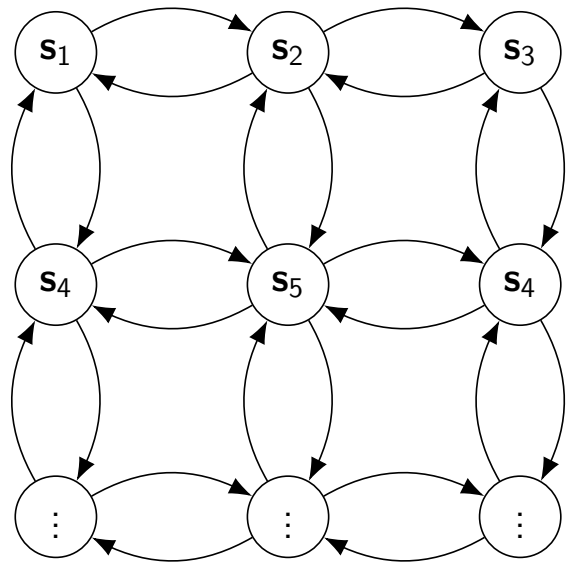
$\mathbf{A}_t = (\mathbf{A}_t^{(\mathrm{A})}, \mathbf{A}_t^{(\mathrm{D})})$ are the actions.



# State Space

- Each $i \in \mathcal{V}$ has a state

$$\boldsymbol{v}_{i,t} = (\underbrace{v_{t,i}^{(\mathrm{Z})}}_{\mathrm{D}}, \underbrace{v_{t,i}^{(\mathrm{I})}, v_{t,i}^{(\mathrm{R})}}_{\mathrm{A}})$$




- System state $\mathbf{s}_t = (\mathbf{v}_{t,i})_{i \in \mathcal{V}} \sim \mathbf{S}_t$

- Markovian time-homogeneous dynamics:

$$\mathbf{s}_{t+1} \sim f(\cdot \mid \mathbf{S}_t, \mathbf{A}_t)$$

$\mathbf{A}_t = (\mathbf{A}_t^{(\mathrm{A})}, \mathbf{A}_t^{(\mathrm{D})})$ are the actions.

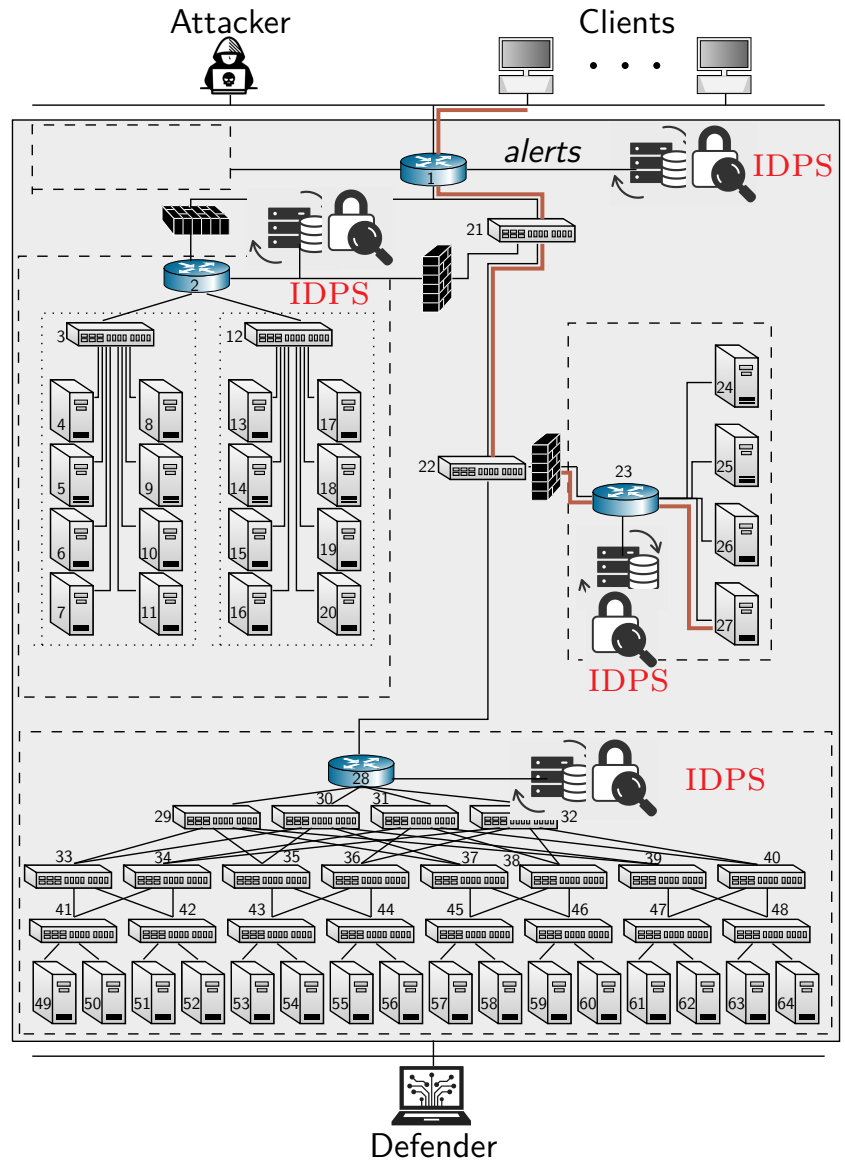
# Observations

- IDPSs inspect network traffic and generate alert vectors:

$$\mathbf{o}_t \triangleq \left(\mathbf{o}_{t,1}, \ldots, \mathbf{o}_{t,|\mathcal{V}|}\right) \in \mathbb{N}_0^{|\mathcal{V}|}$$

$\mathbf{o}_{t,i}$ is the number of alerts related to node $i \in \mathcal{V}$ at time-step $t$.

- $\mathbf{o}_t = (\mathbf{o}_{t,1}, \ldots, \mathbf{o}_{t,|\mathcal{V}|})$ is a realization of the random vector $\mathbf{O}_t$ with joint distribution $Z$



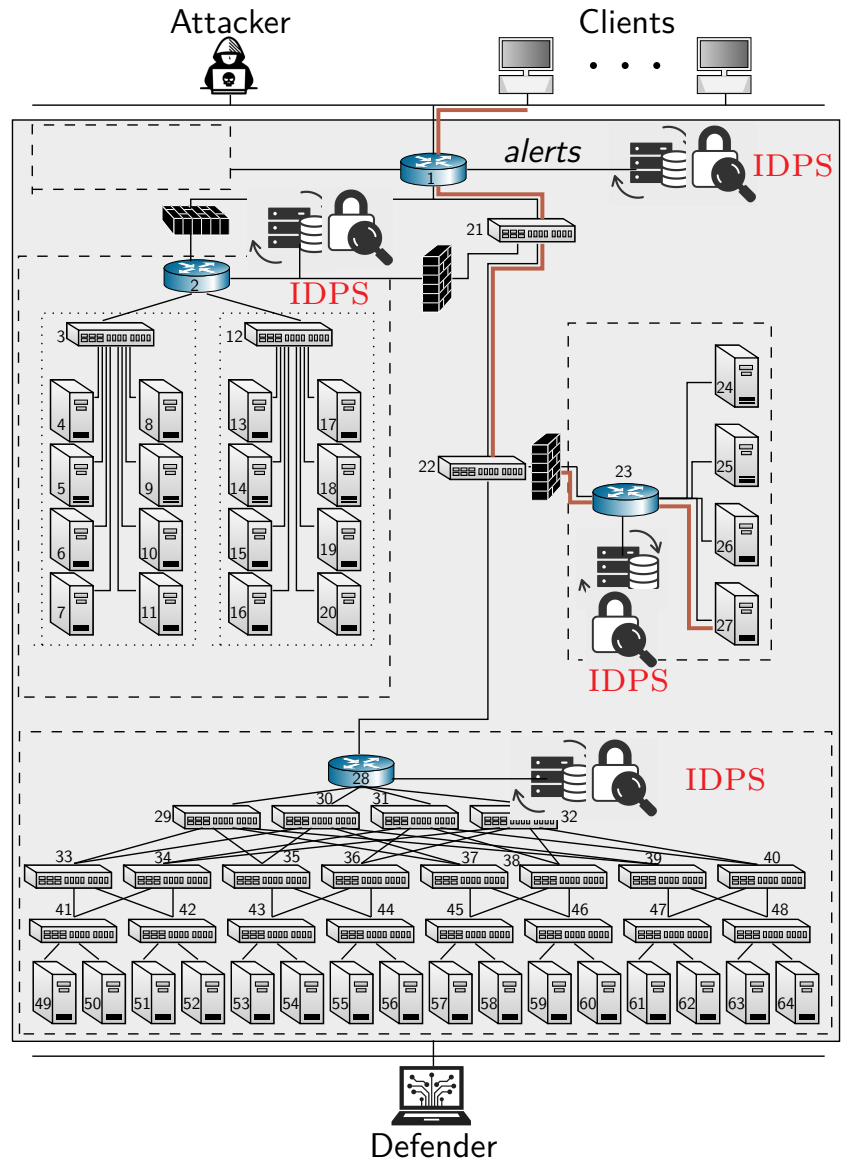
# Observations

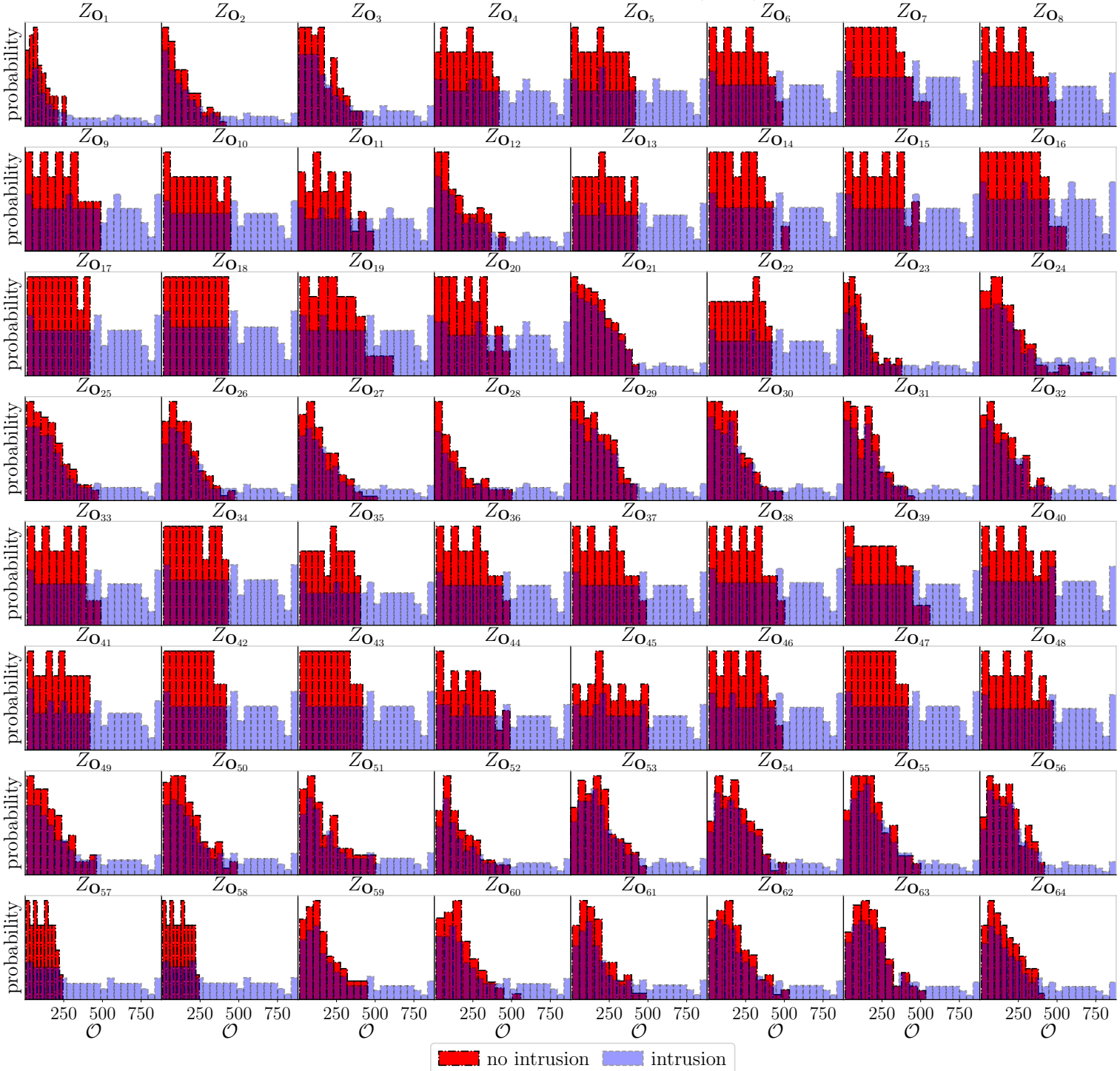
- IDPSs inspect network traffic and generate alert vectors:

$$\mathbf{o}_t \triangleq \left(\mathbf{o}_{t,1}, \ldots, \mathbf{o}_{t,|\mathcal{V}|}\right) \in \mathbb{N}_0^{|\mathcal{V}|}$$

$\mathbf{o}_{t,i}$ is the number of alerts related to node $i \in \mathcal{V}$ at time-step $t$.

- $\mathbf{o}_t = (\mathbf{o}_{t,1}, \ldots, \mathbf{o}_{t,|\mathcal{V}|})$ is a realization of the random vector $\mathbf{O}_t$ with joint distribution $Z$




Distributions of # alerts weighted by priority $Z_{\mathbf{O}_i}(\mathbf{O}_i \mid \mathbf{S}_i^{(\mathrm{D})}, \mathbf{A}_i^{(\mathrm{A})})$ per node $i \in \mathcal{V}$
$Z_{\mathbf{O}_1}$ $Z_{\mathbf{O}_2}$ $Z_{\mathbf{O}_3}$ $Z_{\mathbf{O}_4}$ $Z_{\mathbf{O}_5}$ $Z_{\mathbf{O}_6}$ $Z_{\mathbf{O}_7}$ $Z_{\mathbf{O}_8}$
$Z_{\mathbf{O}_9}$ $Z_{\mathbf{O}_{10}}$ $Z_{\mathbf{O}_{11}}$ $Z_{\mathbf{O}_{12}}$ $Z_{\mathbf{O}_{13}}$ $Z_{\mathbf{O}_{14}}$ $Z_{\mathbf{O}_{15}}$ $Z_{\mathbf{O}_{16}}$
$Z_{\mathbf{O}_{17}}$ $Z_{\mathbf{O}_{18}}$ $Z_{\mathbf{O}_{19}}$ $Z_{\mathbf{O}_{20}}$ $Z_{\mathbf{O}_{21}}$ $Z_{\mathbf{O}_{22}}$ $Z_{\mathbf{O}_{23}}$ $Z_{\mathbf{O}_{24}}$
$Z_{\mathbf{O}_{25}}$ $Z_{\mathbf{O}_{26}}$ $Z_{\mathbf{O}_{27}}$ $Z_{\mathbf{O}_{28}}$ $Z_{\mathbf{O}_{29}}$ $Z_{\mathbf{O}_{30}}$ $Z_{\mathbf{O}_{31}}$ $Z_{\mathbf{O}_{32}}$
$Z_{\mathbf{O}_{33}}$ $Z_{\mathbf{O}_{34}}$ $Z_{\mathbf{O}_{35}}$ $Z_{\mathbf{O}_{36}}$ $Z_{\mathbf{O}_{37}}$ $Z_{\mathbf{O}_{38}}$ $Z_{\mathbf{O}_{39}}$ $Z_{\mathbf{O}_{40}}$
$Z_{\mathbf{O}_{41}}$ $Z_{\mathbf{O}_{42}}$ $Z_{\mathbf{O}_{43}}$ $Z_{\mathbf{O}_{44}}$ $Z_{\mathbf{O}_{45}}$ $Z_{\mathbf{O}_{46}}$ $Z_{\mathbf{O}_{47}}$ $Z_{\mathbf{O}_{48}}$
$Z_{\mathbf{O}_{49}}$ $Z_{\mathbf{O}_{50}}$ $Z_{\mathbf{O}_{51}}$ $Z_{\mathbf{O}_{52}}$ $Z_{\mathbf{O}_{53}}$ $Z_{\mathbf{O}_{54}}$ $Z_{\mathbf{O}_{55}}$ $Z_{\mathbf{O}_{56}}$
$Z_{\mathbf{O}_{57}}$ $Z_{\mathbf{O}_{58}}$ $Z_{\mathbf{O}_{59}}$ $Z_{\mathbf{O}_{60}}$ $Z_{\mathbf{O}_{61}}$ $Z_{\mathbf{O}_{62}}$ $Z_{\mathbf{O}_{63}}$ $Z_{\mathbf{O}_{64}}$
probability
250 500 750
$\mathcal{O}$
no intrusion
intrusion

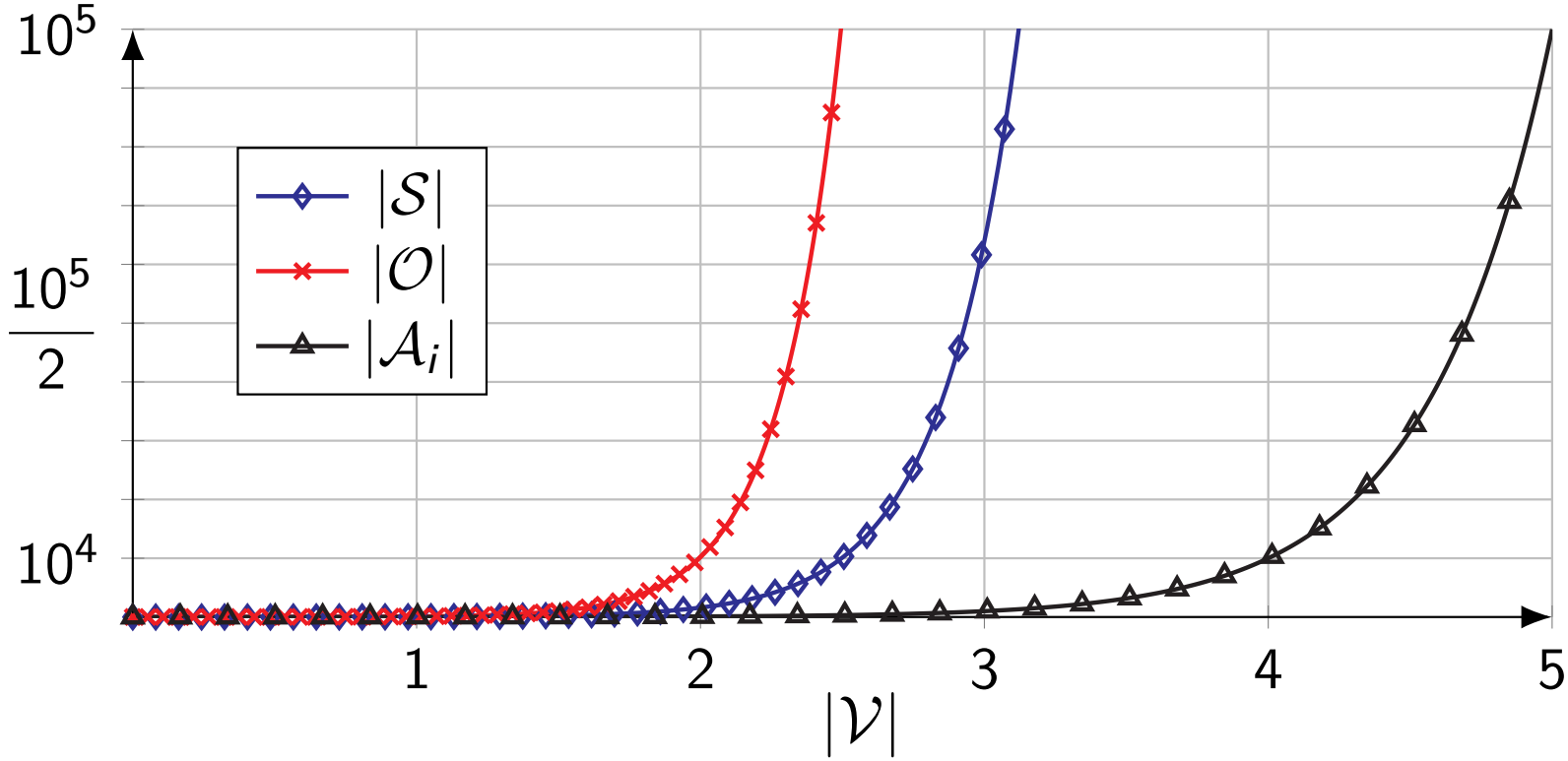
# The (General) Intrusion Response Problem

$$\text{maximize}_{\pi_{\mathrm{D}} \in \Pi_{\mathrm{D}}} \ \text{minimize}_{\pi_{\mathrm{A}} \in \Pi_{\mathrm{A}}} \ \mathbb{E}_{(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})}[J]$$

$\mathbb{E}_{(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})}$ denotes the expectation of the random vectors $(\mathbf{S}_t, \mathbf{O}_t, \mathbf{A}_t)_{t \in \{1,\ldots,T\}}$ when following the strategy profile $(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})$

# The Curse of Dimensionality

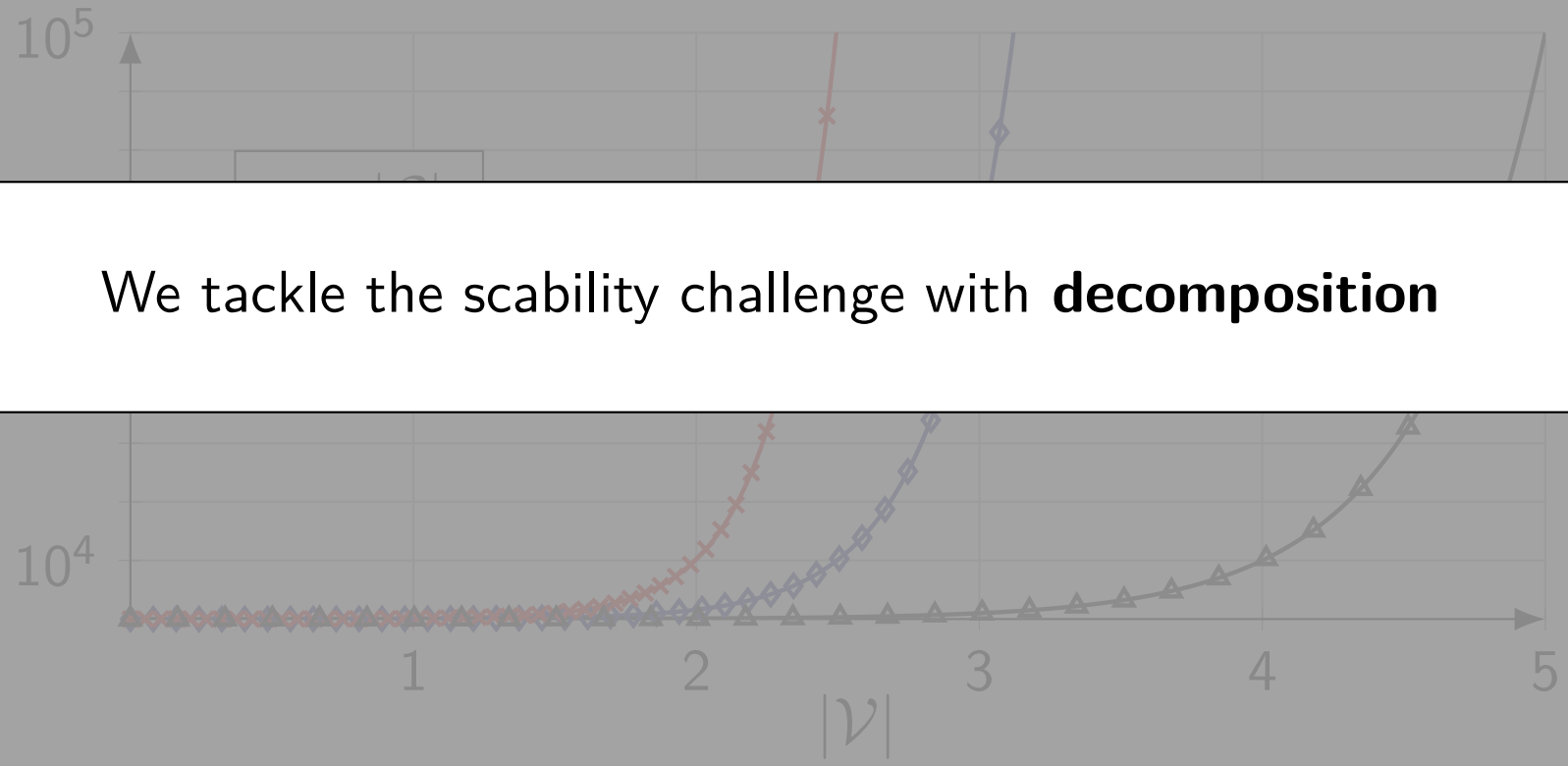
- Solving the game is computationally intractable. The state, action, and observation spaces of the game **grow exponentially** with $|\mathcal{V}|$.




Growth of $|\mathcal{S}|$, $|\mathcal{O}|$, and $|\mathcal{A}_i|$ in function of the number of nodes $|\mathcal{V}|$

# The Curse of Dimensionality

- While (1) has a solution (i.e the game $\Gamma$ has a value (Thm 1)), computing it is intractable since the state, action, and observation spaces of the game **grow exponentially** with $|\mathcal{V}|$.

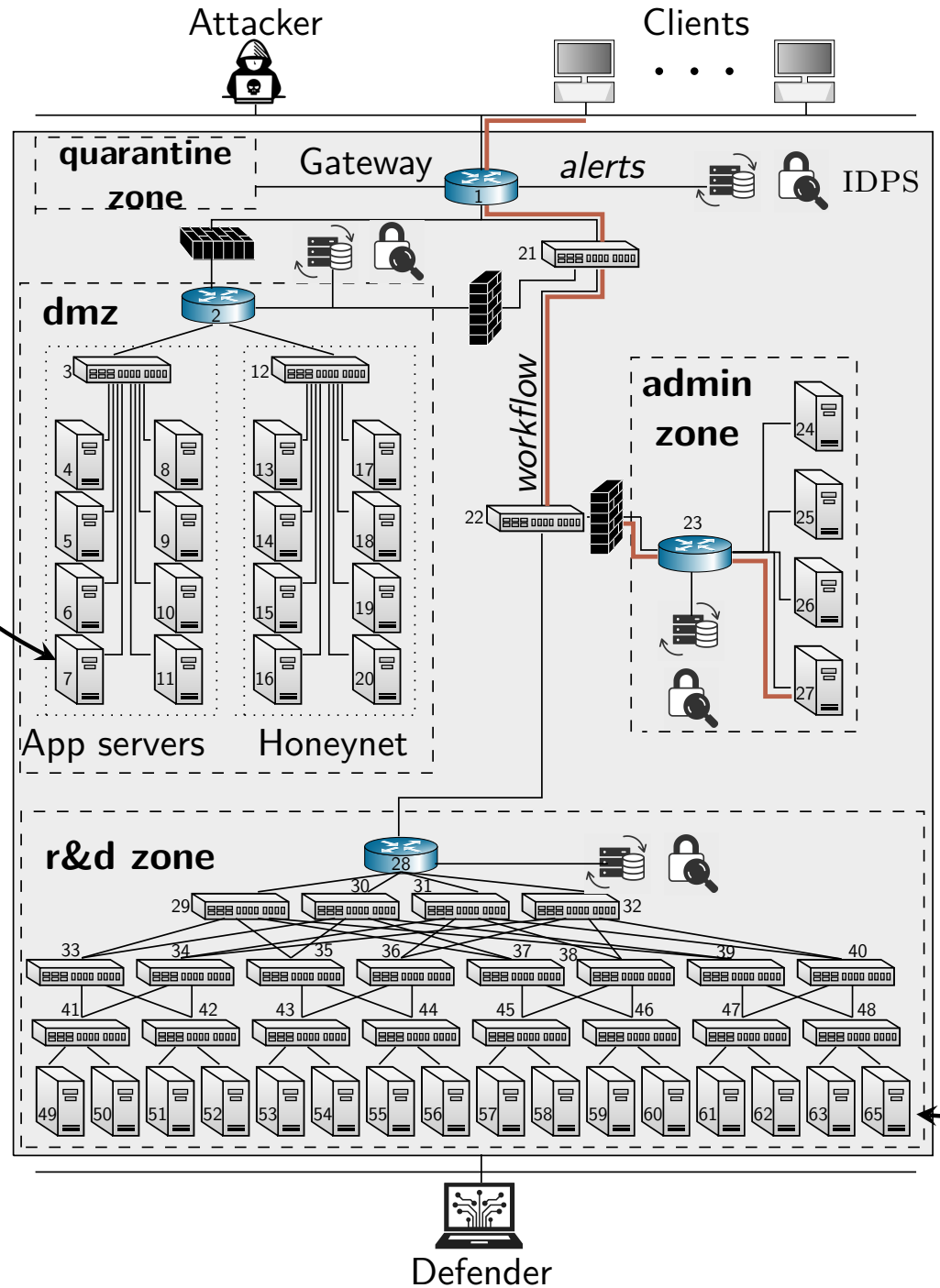



Growth of $|\mathcal{S}|$, $|\mathcal{O}|$, and $|\mathcal{A}_i|$ in function of the number of nodes $|\mathcal{V}|$

We tackle the scability challenge with **decomposition**

# Intuitively..

The optimal action here...



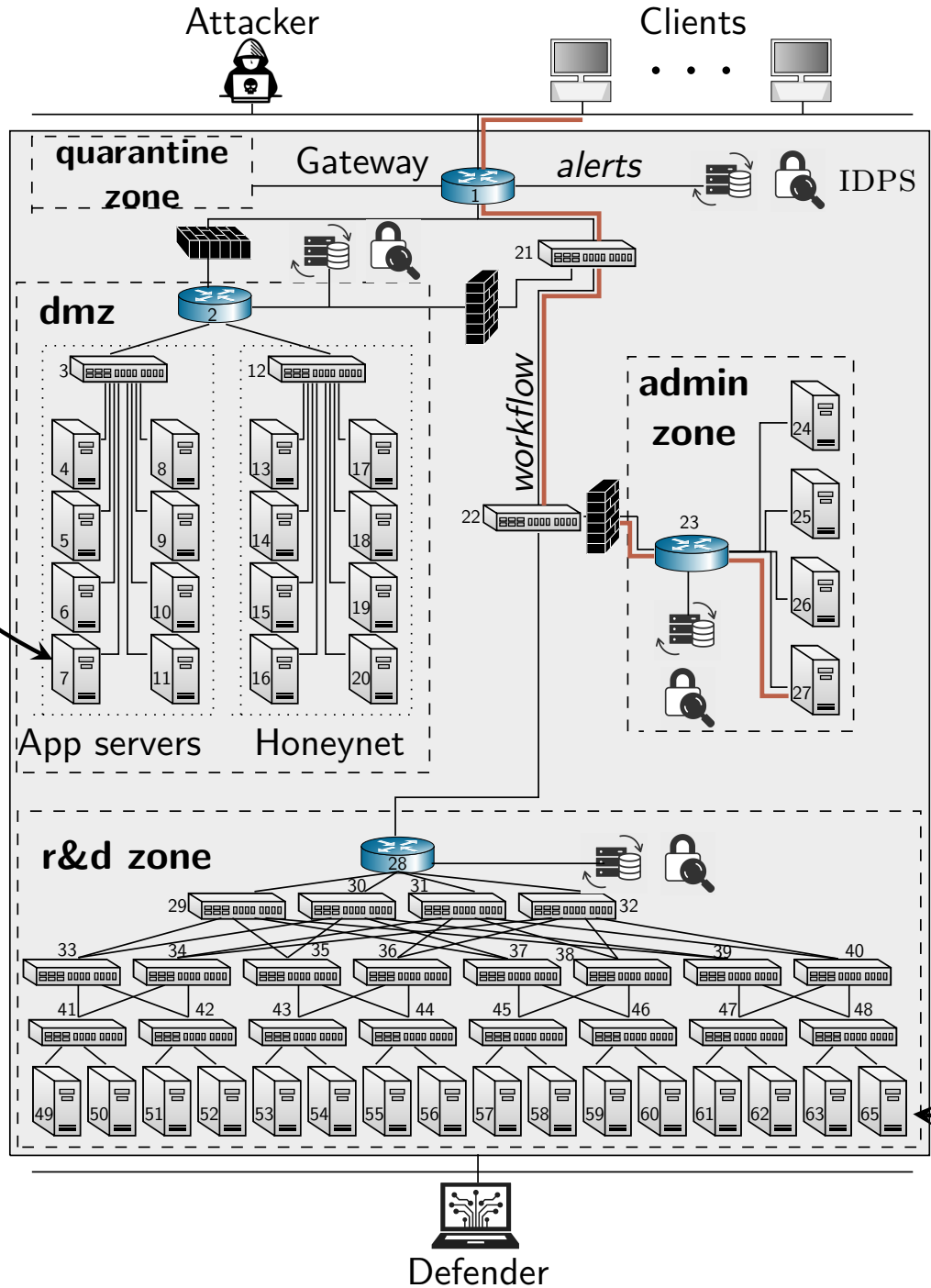

Does not directly depend on the state or action of a node down here

# Intuitively..

The optimal action here...

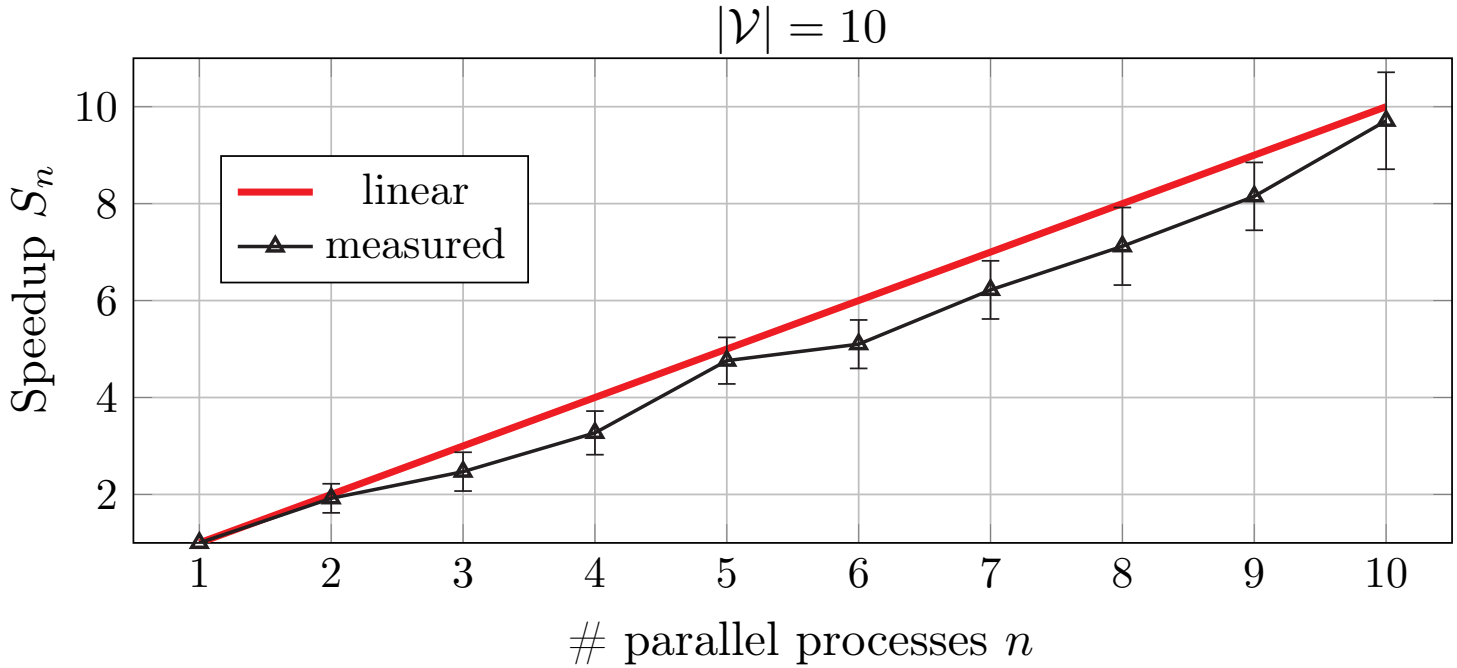
But they are not completely independent either.

How can we exploit this structure?



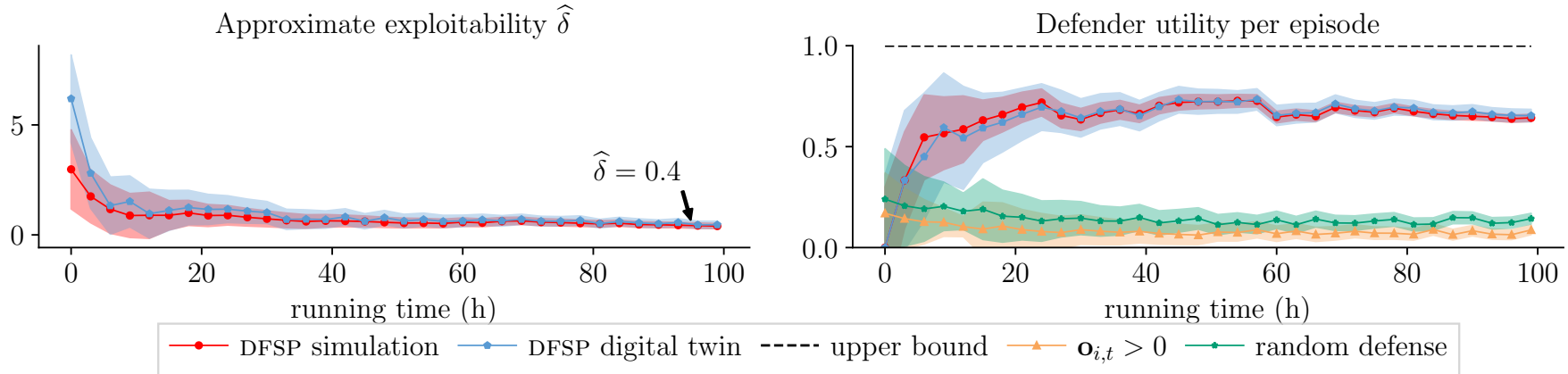

Does not directly depend on the state or action of a node down here

# Scalable Learning through Decomposition



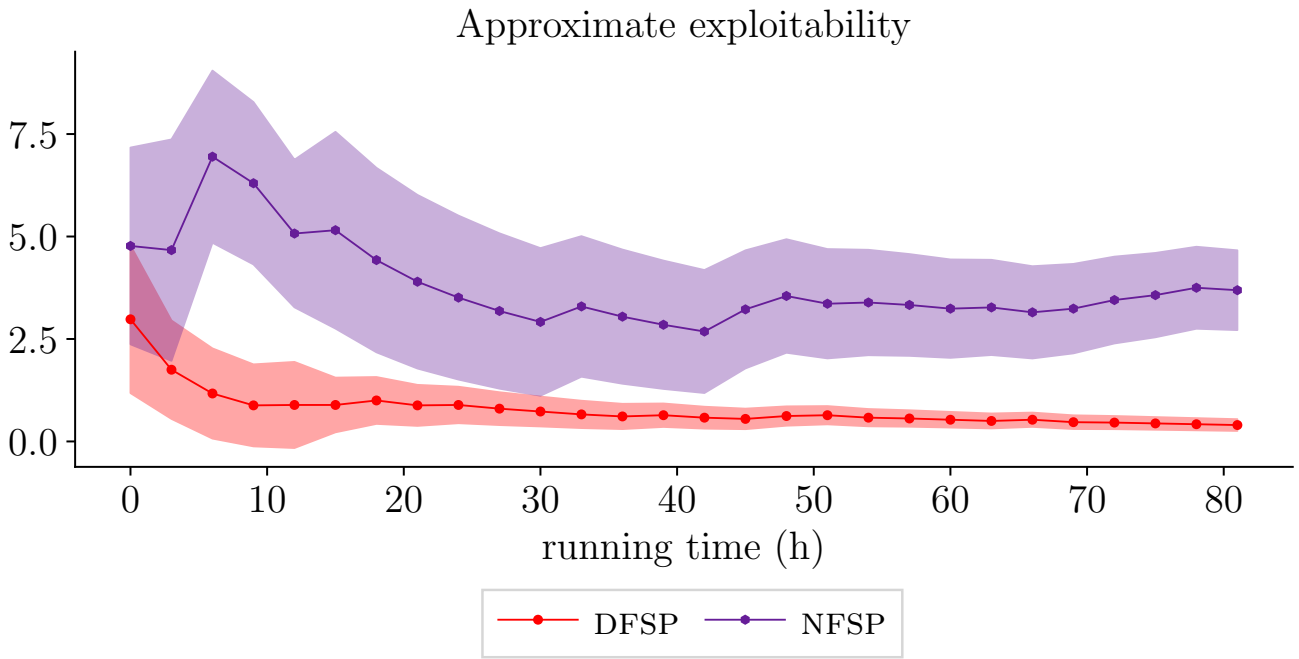

**Speedup of best response computation** for the decomposed game; $T_n$ denotes the completion time with $n$ processes; the speedup is calculated as $S_n = \frac{T_1}{T_n}$; the error bars indicate standard deviations from 3 measurements.

# Learning Equilibrium Strategies




Learning curves obtained during training of DFSP to find optimal (equilibrium) strategies in the intrusion response game; **red and blue curves relate to dfsp**; black, orange and green curves relate to baselines.

# Comparison with NFSP




Learning curves obtained during training of DFSP and NFSP to find optimal (equilibrium) strategies in the intrusion response game; **the red curve relate to dfsp** and the purple curve relate to NFSP; all curves show simulation results.

# Conclusions

- We develop a *framework* to automatically learn security strategies.
- We apply the framework to an **intrusion response use case**.
- We derive properties of optimal security strategies.
- We evaluate strategies on a **digital twin**.
- Questions $\rightarrow$ demonstration