

Introduction

Industrial Control Systems (ICS) run critical infrastructure and are increasingly interconnected, exposing new attack surfaces and vectors for malicious actors to exploit. Penetration testing is among the principal tools used to uncover misconfigurations and viable attack paths. In this work we aim to use Reinforcement Learning (RL) as a way to automate pentesting for ICS networks. Yet **no framework exists for training RL pentesting agents on ICS**:

- **No ICS modelling** — RL pentest environments ignore ICS TTPs, protocols, and device semantics; Currently available ICS cyber-ranges (SCEPTRE) are unsuited for RL training.
- **Throughput** — Large ICS networks need high throughput simulators to train useful RL policies.
- **Reality gap** — simulation-trained policies need validation on realistic deployments to be usable in the real-world.

ICS Sim2Emu adapts the high-throughput simulator NASimJax to ICS and adds an emulation layer validating agents on realistic ICS networks. The agent learns *attack-path discovery*: chaining reconnaissance, exploitation, and privilege escalation across hosts whose roles and services are initially hidden (Fig. 1).

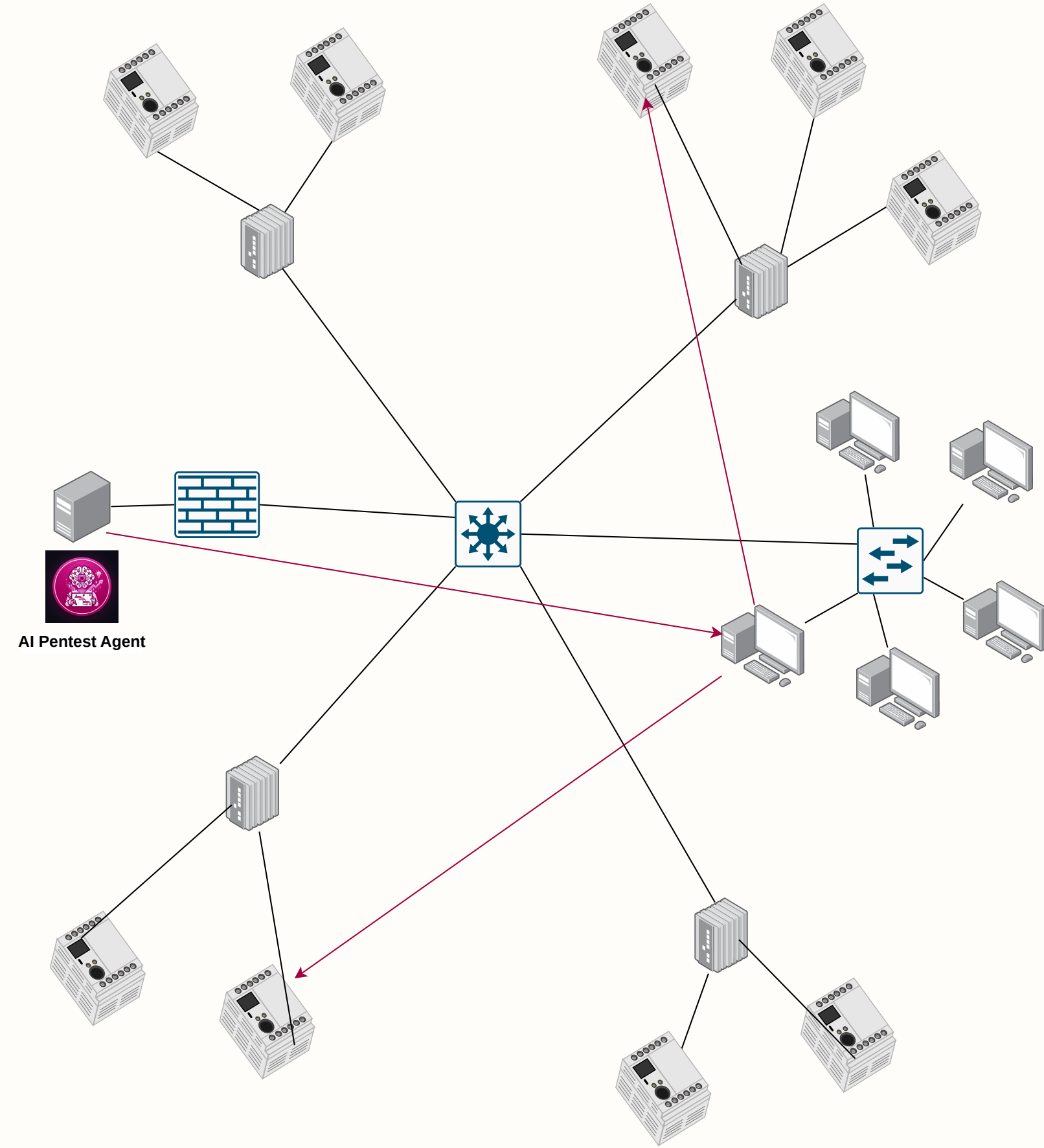


Fig. 1: Attack-path discovery across an ICS network.

ICS Sim2Emu

From NASimJax we inherit the Contextual-POMDP formulation, state representation, and contextual training loop, and add ICS TTPs plus validation on real infrastructure, also specializing the network generator to ICS setting. The simulator is pure JAX (JIT-compiled, vectorised rollouts), reaching $\sim 100\times$ compared to prior simulators.

Interface invariance. Both simulation and emulation expose an *identical* action set and observation structure, so a simulation-trained policy runs in emulation *without retraining*. An *Action Translation Module* maps each abstract action to a concrete script invocation; a *State Translation Module* rebuilds the state from live responses (Fig. 2).

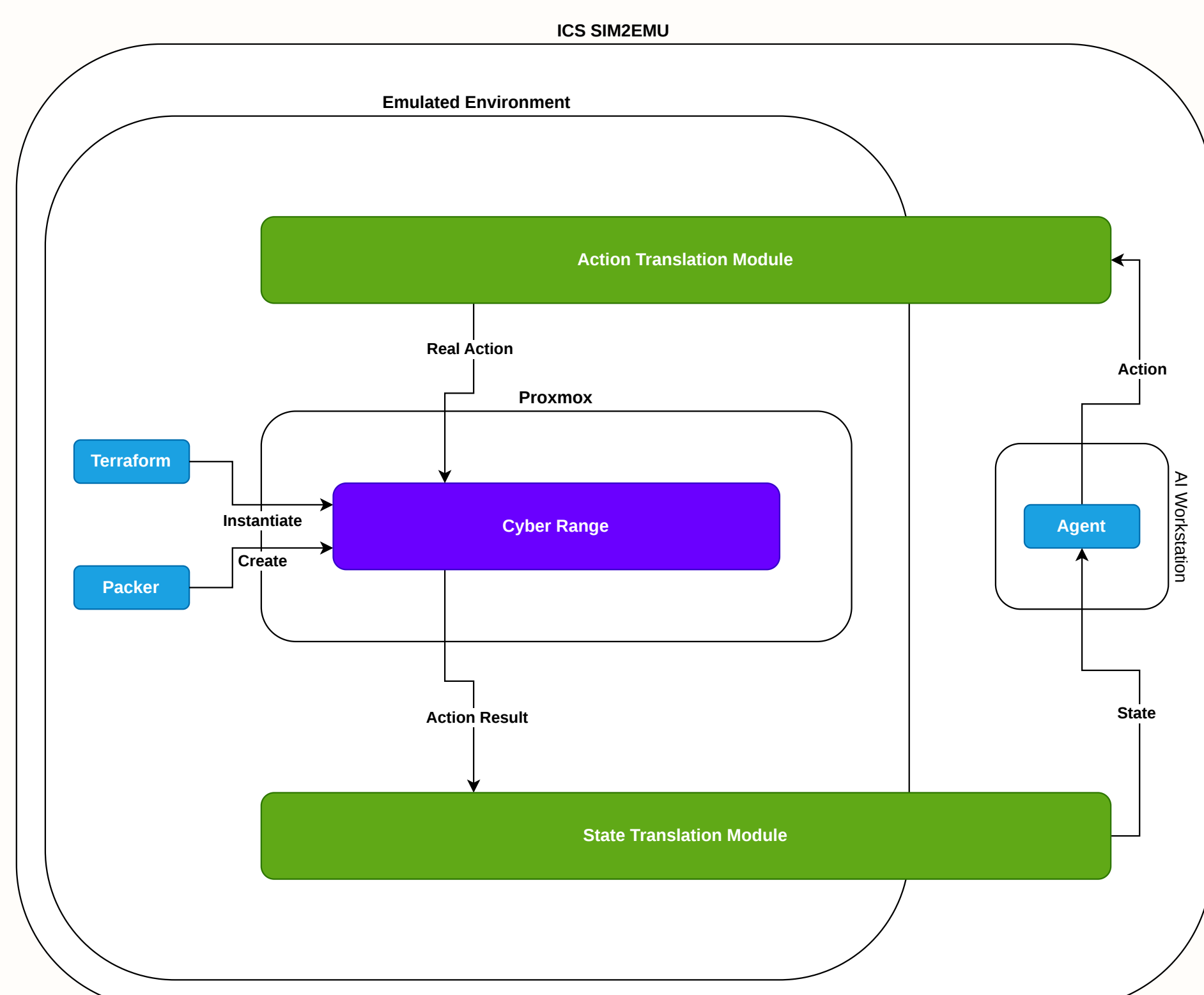


Fig. 2: ICS Sim2Emu architecture overview.

Infrastructure-as-Code. Proxmox, Packer, and Terraform make scenarios reproducible, fast to deploy, and easy to extend.

ICS Network Generator

- Built on the **Purdue Reference Model**: Level 3 (operations) down to Level 1 (control).
- Assembles scenarios from configurable templates; each host's role (PLC, engineering workstation, log archive, ...) fixes its services and its position in the topology.
- **Domain randomization**: every episode is a freshly sampled network, so the policy must generalise across ICS structures rather than memorise a single layout.

Attack Scenarios

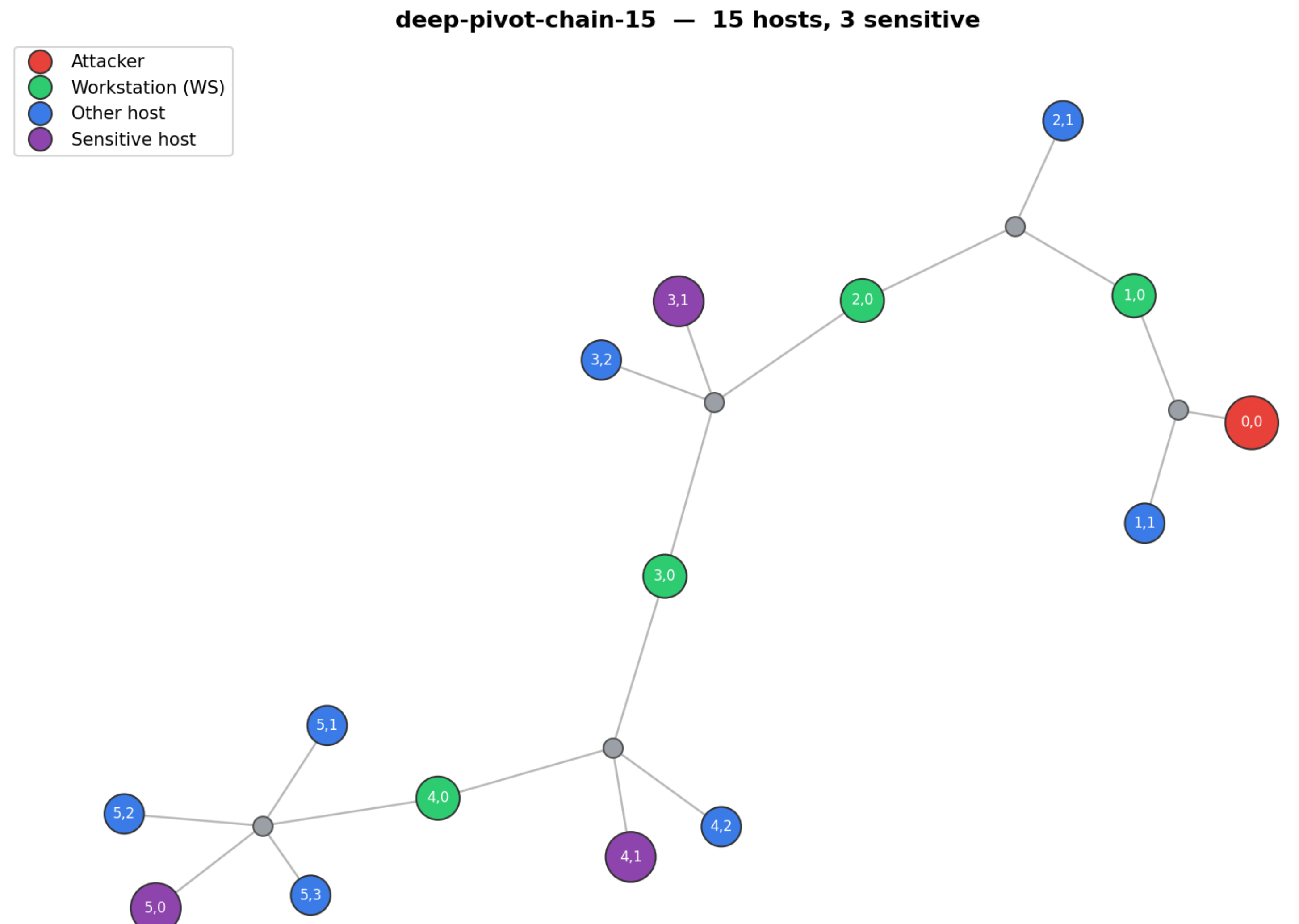


Fig. 3: Example scenario — deep-pivot-chain-15 (15 hosts, 6 subnets).

Experimental Evaluation

Three questions: (i) sim-to-emu transfer without retraining? (ii) RL-algorithm behaviour? (iii) generalisation to unseen networks? We compare **PPO** (on-policy) and **PQN** (value-based) under domain randomization (Bayesian hyperparameter search, ≥ 250 trials each).

Key findings:

- **Exact sim-to-emu transfer**: the agent reproduces the identical attack path; observation, state, and reward match at every step.
- **Zero-shot transfer**: one policy solves all five out of distribution scenarios (PPO).
- **Depth separates them**: on the deepest pivots (deep-pivot-chain-15, dmz-pivot-15) PPO stays reliable while PQN fails or is high-variance (Fig. 4).

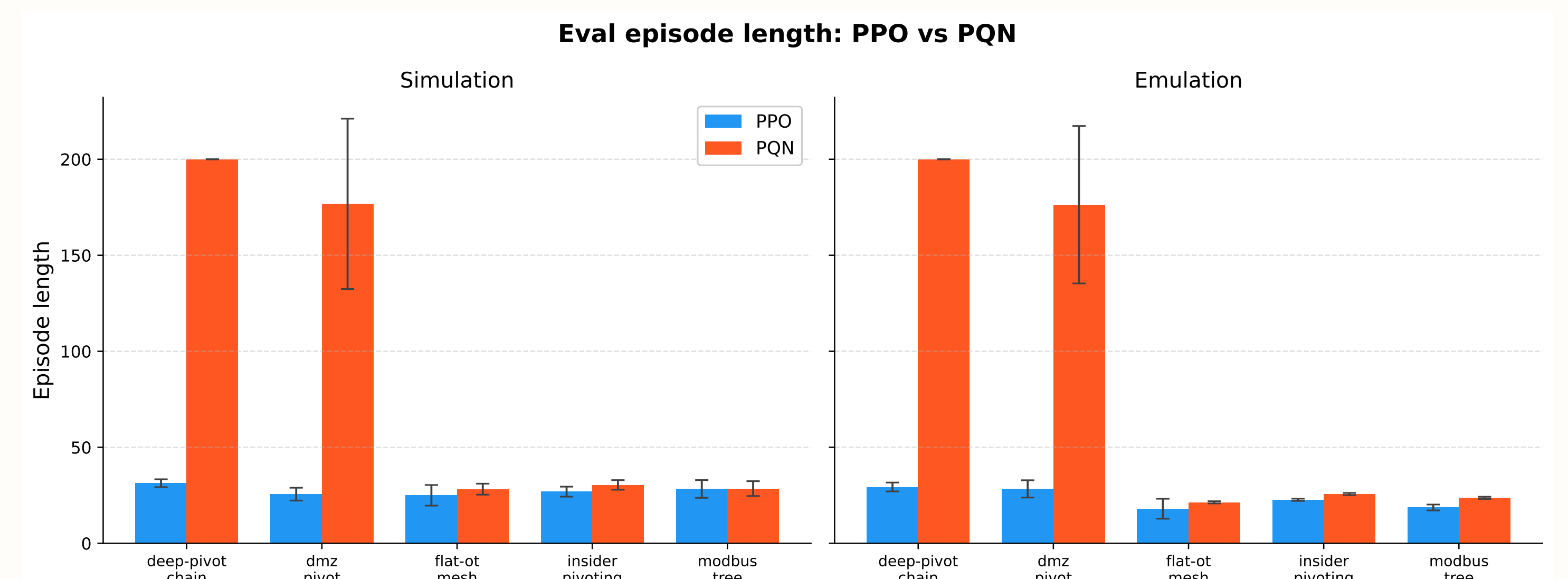


Fig. 4: Evaluation episode length — PPO vs PQN, in simulation and emulation.

What this platform unlocks

- One of the **first** RL environment modelling ICS TTPs and protocols, with sim-to-real validation on emulated infrastructure.
- **Reproducible ICS testbeds** via industry-standard Infrastructure-as-Code.
- Enables further study of RL generalization and robustness in the context of ICS scenario models, with real-world validation capacity.