Automated Cyber Range Infrastructures for Securing Tomorrow: Lessons from Range42

Abstract

The future of cyber conflict demands training and testing platforms that anticipate tomorrow's threats rather than replicate yesterday's incidents. This paper presents Range42, an open cyber range developed by the National Cybersecurity Competence Center (NC3) under the Luxembourg House of Cybersecurity together with DIGISQUAD. Designed as a flexible, automated platform, Range42 lowers barriers to building realistic and reproducible training environments while fostering collaborative research and development.

Our research introduces two core contributions and a future standard. First, we describe an extensible catalog system of vulnerable and misconfigured environments. With over 100 curated CVEs and misconfigurations identified and approximately 20 already deployable, this system enables structured reproduction of real-world attack surfaces for controlled experiments. Second, we detail an orchestration framework capable of deploying multi-subnet infrastructures, making it possible to simulate complex enterprise-grade networks with isolation and fidelity. Finally we want to develop a scenario standard language for cyber ranges, well knowing that quite a few efforts already exist yet none englobes the entire process of a cyber range exercise. We highlight a proof-of-concept LLM-assisted workflow: prompt → scenario narrative + schema → validation → ingestion; this enables fast, potentially innovative and always up to date on the current threat landscape and the most common attacks.

We propose range42-catalog, a modular, open catalog of vulnerable and misconfigured environments, designed for extensibility, interoperability, and community-driven growth. The catalog aligns with our Ansible-based architecture and supports declarative scenario assembly across tools.

We also present a scenario description standard (dual human-readable + machine-ingestible schema) for defining cyber range scenarios (including technical and non-technical injects), enabling a single definition to generate JSON for deployment. We highlight a proof-of-concept LLM-assisted workflow: prompt → scenario narrative + schema → validation → ingestion.

We outline the automation pipeline (Proxmox integration, Ansible-driven orchestration, container/VM deployment, Tailscale-based zero-trust connectivity), and evaluate catalog ingestion, scenario generation, and deployment consistency. We conclude that an open, standardized catalog + scenario architecture is key to "securing tomorrow" through shared scenario ecosystems, faster tool integration, and collaborative innovation.

1 Introduction

Cyber threats evolve faster than training and evaluation methodologies. To secure tomorrow, cyber ranges must enable realistic, reproducible, and rapidly composed environments that reflect emerging attack surfaces and defensive strategies. We present Range42, an open cyber range platform emphasizing automation, flexibility, and collaboration. Our contributions are: (1) an

extensible catalog of vulnerable/misconfigured systems for research-grade reproducibility; and (2) an orchestration framework for deploying multi-subnet infrastructures approximating enterprise topologies. We outline our automation pipeline and report lessons for scalability, isolation, and openness. Further a standard scenario description that blends human readability and machine ingestion, enabling consistent narrative and technical deployment definitions is under consideration. We also explore an experimental LLM-assisted pipeline to generate scenarios from prompts; this might be interesting for full automation information gathering. Playing hundreds of generated scenario to extract the generated telemetry might be an interesting source of data. Especially when paired with real-world events and digital twins of certain breaches.

2 Related Work

Existing cyber range catalogs or scenario repositories tend to be closed, bespoke, or tightly coupled to particular platforms. Some efforts provide deployment DSLs or scripts, but few support a human-editable narrative plus structured schema, or offer APIs for third-party tooling. We review virtual scenario description languages, shared scenario repositories, and limitations in openness and interoperability.

3 System Architecture & Catalog Design

Range42 operates via Proxmox, Ansible, and container/VM integration. The range42-catalog is structured as a modular repository of scenario components (e.g. vulnerable hosts, network topologies, inject modules), each described with metadata (e.g. prerequisites, dependencies, scoring hooks). We maintain extension APIs so external tools can list, validate, augment, or contribute entries. The catalog integrates seamlessly with Ansible roles to map component definitions to deployment logic.

4 Vulnerability and Misconfiguration Catalog

We curate an inventory of $\sim$100 CVEs/misconfigurations (with $\sim$20 currently deployable), using build descriptors and snapshotting to balance reproducibility with support for proprietary or atypical systems. The design supports traceability, variant creation, and controlled risk exposure during exercises.

5 Scenario Description Standard

We define a domain-specific schema (YAML / JSON hybrid) for scenario narratives. A scenario file includes sections like: background story, objectives, inject schedule, scoring rules, triggers, technical blueprint. From this human-readable form, we generate canonical machine-readable JSON which Range42 ingests to deploy and run the scenario. We include validation logic to catch schema errors or dependency mismatches.

6 Integration & Deployment Pipeline

From catalog + scenario schema to live lab: we describe the orchestration path (Ansible playbooks, Proxmox API calls, network wiring, Tailscale connectivity). We discuss how catalog metadata guides resource allocation, dependency resolution, and runtime coordination of injects and events.

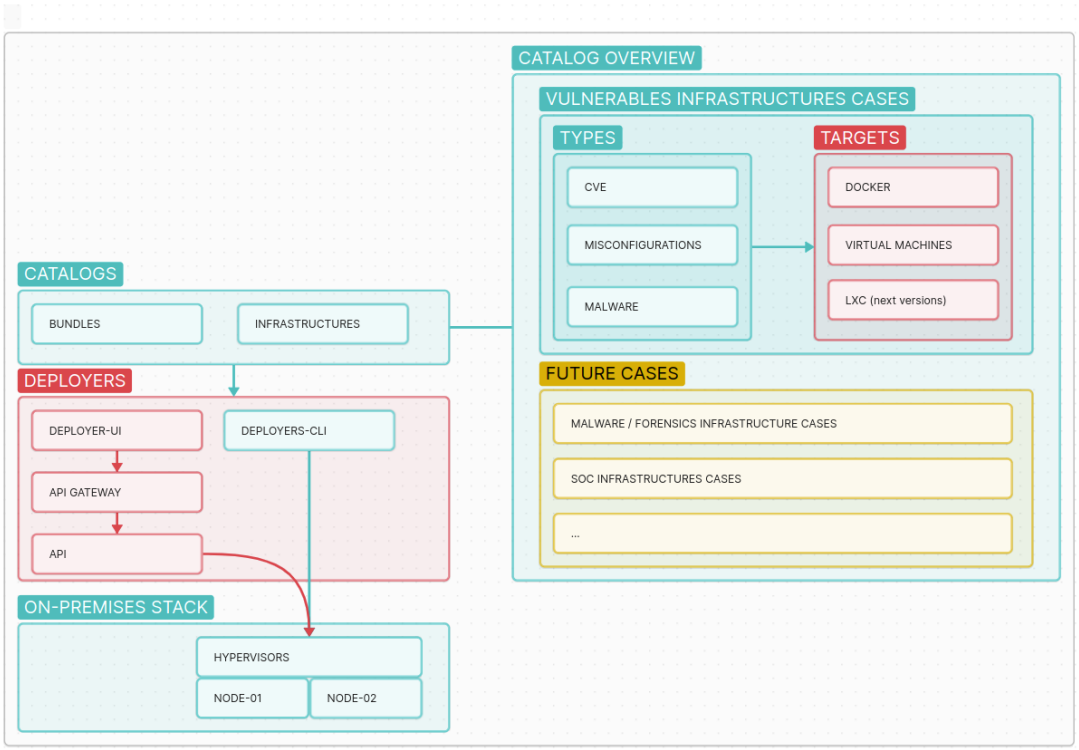

Figure 1: High-level Range42 catalog architecture.

7 LLM-Assisted Scenario Generation

We built a proof-of-concept workflow: given a prompt (e.g. "simulate a ransomware attack in 3-subnet corporate network"), an LLM produces a draft narrative + schema. We feed that into validation logic, correct or flag inconsistencies, then ingest it into the catalog for deployment. Preliminary results and failure modes are discussed.

8 Evaluation

We measure catalog ingestion success rate, scenario consistency (i.e. narrative vs deployed topology), deployment latency, and reusability across runs. Where full metrics are limited, we include qualitative feedback from instructors and early users.

9 Discussion

We reflect on expressivity vs. enforceable constraints, schema versioning, contribution governance, catalog federation across platforms, and operational challenges (e.g. schema drift, extension conflicts). The choice of "catalog" (versus "inventory") is rooted in the alignment with Ansible's modular, declarative paradigm, and the desire for interoperability.

10 Future Work

Expand the catalog with richer scenarios (forensics, social engineering, insider threats), build GUI editors for the standard format, integrate stronger LLM-based validation and correction, and establish federated catalogs across multiple range platforms with shared schemas. Advancing a visual lab designer; integrating malware analysis/forensics workflows; and supporting richer hybrid topologies.

11 Conclusion

By combining an open catalog architecture with a standardized, dual-mode scenario description and LLM-assisted generation, Range42 is positioned not just as a cyber range but as part of an ecosystem of interoperable scenario tooling. In this context we try to follow the lead of the MISP project when it comes to community building, information sharing, open source ethos and innovation by having open standards. We believe this approach will help catalyze collaborative, scalable, and future-ready cyber range research and training. Automated ranges like Range42 help secure tomorrow by enabling hands-on research and training humans as well as generating data for model generation at scale.

References

[1] Proxmox VE. Available: https://www.proxmox.com/

[2] Ansible Documentation. Available: https://docs.ansible.com/

[3] Docker Documentation. Available: https://docs.docker.com/

[4] Tailscale Documentation. Available: https://tailscale.com/kb/

[5] IEEE Editorial Style Manual. Available: https://journals.ieeeauthorcenter.ieee.org/