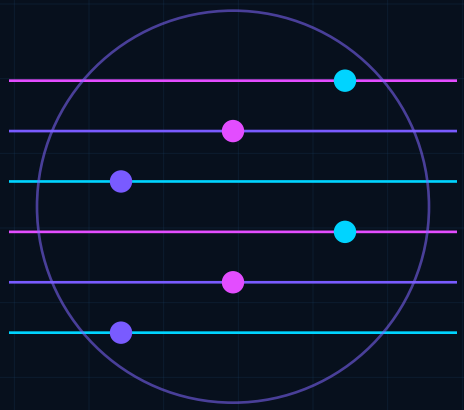
NETWORK SIMULATION AND DIGITAL TWINS

Network Digital Twin Limitations (Batfish/Containerlabs constraints)

Current-source review, evidence-bounded adoption decision, explicit limitations, reproducibility controls, and preserved source research note.



PERMANENT ID	EVIDENCE	ADOPTION	FRESHNESS
RES-005	A-	CONDITIONAL ADOPTION	STABLE WATCH

Freshness review: 2026-09-17 / Next scheduled review: 2027-03-16

RESEARCH CONTROL SURFACE

Required Research Fields

Each field remains independently reviewable; evidence strength does not imply production authority.

EVIDENCE GRADE — **A-**

Strength of the retained source set and technical basis.

SOURCE AUTHORITY — **3 registered authorities**

Batfish Documentation; Containerlab Documentation; RFC 8342 - Network Management Datastore Architecture (NMDA)

FRESHNESS — **STABLE WATCH**

Reviewed 2026-09-17; dynamic sources require event-driven revalidation.

CONFLICTS — **VISIBLE / NOT SILENT**

Differences between specification, vendor behavior, research claims, and reproduced evidence must remain explicit.

LIMITATIONS — **EXPLICIT**

No certification, procurement approval, legal conclusion, or unbounded production claim is created by this report.

REPRODUCIBILITY — **REQUIRED**

Material claims require exact versions, representative data, failure cases, artifacts, and repeatable acceptance criteria.

ADOPTION POSTURE — **CONDITIONAL ADOPTION**

Adopt Batfish and Containerlab as complementary analysis and lab components, not as complete digital twins. Require explicit fidelity statements, unsupported-feature registers, and observed-state validation.

REVIEW TRIGGER — **2027-03-16**

Scheduled at 180 days; earlier on material source, vulnerability, implementation, or contradictory-evidence change.

SOURCE AUTHORITY / FRESHNESS

Authority Register and Freshness Delta

Primary-source lineage is preserved; current-source changes are separated from unperformed implementation claims.

AUTHORITY 01

Batfish Documentation

https://batfish.readthedocs.io/

AUTHORITY 02

Containerlab Documentation

https://containerlab.dev/manual/

AUTHORITY 03

RFC 8342 - Network Management Datastore Architecture (NMDA)

https://www.rfc-editor.org/rfc/rfc8342

2026-09-17 FRESHNESS DELTA

Primary-source register retained and freshness controls advanced to the current review date. No new product or laboratory performance claim is introduced without reproduced evidence.

Freshness policy: scheduled review + immediate review on source supersession, material release, vulnerability, retraction, or contradictory evidence.

Evidence Must Survive Contact With Reality

- Source authority is not implementation evidence. Product behavior, scale, security, interoperability, and operator fit remain environment-specific.
- Conflicting evidence is recorded rather than averaged away. Specification behavior, vendor implementation, preprint results, and production observations are separate evidence classes.
- Quantitative claims are informative until reproduced under documented hardware, software, data, topology, load, and failure conditions.
- Adoption posture is reversible. A stronger source set can coexist with a pilot-only or research-only recommendation when maturity, safety, economics, or operability remain unresolved.

ADOPTION / VALIDATION / REVIEW

Decision Gate and Validation Plan

ADOPTION POSTURE

CONDITIONAL ADOPTION

Adopt Batfish and Containerlab as complementary analysis and lab components, not as complete digital twins. Require explicit fidelity statements, unsupported-feature registers, and observed-state validation.

1 / SOURCE	Pin exact versions, publication state, retrieved artifacts, and source hashes.
2 / PROTOTYPE	Demonstrate the core mechanism with representative data and instrumented execution.
3 / FAILURE	Exercise malformed input, dependency loss, stale state, overload, recovery, rollback, and misuse.
4 / BASELINE	Compare against an incumbent, classical, or simpler architecture using the same workload.
5 / ACCEPT	Record acceptance criteria, residual limitations, owner, expiration, and revalidation triggers.

EXPIRATION / REVIEW TRIGGER

Next scheduled review: 2027-03-16 (180-day cadence). Review sooner after material source revision, breaking implementation release, critical vulnerability, retraction, benchmark contradiction, changed workload, or changed legal/safety boundary.

8. Complete Source Research Note

SOURCE-LINEAGE NOTICE

The following material preserves the complete original source note, excluding only the conversational wrapper and outer Markdown fence. Legacy status labels and absolute language are retained for traceability and do not override the candidate status, limitations, or adoption decision in this edition.

Document ID: RES-005 Version: 1.0.0 (Definitive Registry Edition) Status: Published Research Note Classification: Public Organization: Mythos Systems (MYTHOS AI) Owner: Aaron Stovall Effective Date: 2026-07-16 Applies To: Network architects, SREs, and DevOps engineers designing pre-deployment validation pipelines, formal verification logic, and network simulation environments Companion Standards: MYTHOS-NET-0003 (Network Digital Twin & Simulation Verification), MYTHOS-NET-0001 (Network Architecture), MYTHOS-SWE-0007 (Formal Verification)

The architecture of network validation has embraced the "Digital Twin" as a silver bullet. Organizations parse configurations through Batfish or spin up topologies in Containerlab, and if the simulation returns a green checkmark, they declare the change safe for production. This is a dangerous fallacy.

A digital twin is not a clone of reality; it is a mathematical abstraction. By definition, abstractions omit details. When those omitted details include ASIC forwarding behavior, hardware TCAM limits, or dynamic protocol convergence physics, the simulation will pass while the production network crashes.

This research note defines the architectural limitations of network digital twins. It dissects the static analysis constraints of Batfish (parser blind spots, lack of dynamic state) and the control-plane/data-plane gaps of Containerlab (ASIC emulation failures, physical layer blindness). Understanding these constraints is a prerequisite for building the multi-layered verification pipelines mandated by the Mythos Network Standard.

To understand why twins fail, we must categorize what they are actually modeling.

- Configuration Analysis (Batfish): Parses raw text configs into a vendor-agnostic data model (private AST) and mathematically calculates forwarding tables. It does not run the network.
- Control-Plane Simulation (Containerlab/cEOS): Runs actual vendor OS images in Docker containers. The routing protocols form real adjacencies and calculate real RIBs/FIBs.
- The Missing Piece (Data-Plane / Physical Reality): Neither tool accurately models what happens when a 100Gbps traffic stream hits a hardware ASIC with limited TCAM space, or when a dirty fiber connection causes CRC errors.

When an organization treats the Configuration or Control-Plane twin as a complete representation of reality, they fall victim to the Simulation Blind Spot.

Batfish is exceptional at answering the question: "If I apply these configs, will A be able to reach B?" It does this without running the protocols, relying purely on mathematical graph theory. However, its architectural limitations are severe.

2.1 Parser Fragility

Batfish does not run the vendor OS; it parses the text configuration using reverse-engineered grammars.

- The Limitation: When a vendor (e.g., Cisco IOS-XE, Junos) introduces a new syntax feature, Batfish often does not know how to parse it.
- The Impact: Batfish will silently ignore the unrecognized line. If that line is a crucial security policy (e.g., a new `match` statement in a route-map), the simulation will show a false positive (traffic passes when it should fail, or vice versa) because it deployed an incomplete model.

2.2 Lack of Dynamic State

Batfish calculates a steady-state forwarding table. It assumes all links are up and all protocols have converged.

- The Limitation: It cannot model failure scenarios or convergence dynamics.
- The Impact: Batfish can tell you if a route exists, but it cannot tell you that when Link A fails, OSPF will take 45 seconds to reconverge, causing a catastrophic BGP holddown timer expiration. It is blind to the physics of time and failure.

2.3 Abstracted Forwarding Behavior

Batfish models forwarding as a simple longest-prefix-match (LPM) graph.

- The Limitation: It ignores hardware-level forwarding constraints like ACL TCAM exhaustion, PBR (Policy-Based Routing) processing order on specific ASICs, or hardware encryption overhead.
- The Impact: A config that passes Batfish's reachability assertions may crash a physical switch because the complex ACL exceeds the hardware's TCAM capacity, causing packets to be punted to the CPU and dropped.

Containerlab (and similar tools like GNS3 or EVE-NG) solves Batfish's parser problem by running the actual vendor OS. If the OS accepts the config, the twin accepts it. However, this introduces a different set of physical limitations.

3.1 The Linux Dataplane Illusion

Containerlab runs network OSes (like Arista cEOS or Cisco XRd) as Docker containers. These containers use the Linux kernel for forwarding.

- The Limitation: The Linux kernel is not a hardware ASIC. It does not have the forwarding latency, buffer architecture, or TCAM limits of a physical switch.
- The Impact: A routing policy that relies on hardware-level forwarding (e.g., slicing TCAM for VRFs) will execute fine in Containerlab but may fail or perform drastically differently on physical metal. You are testing the control plane, but relying on a fake data plane.

3.2 Physical Layer Blindness

Containerlab assumes all virtual Ethernet (veth) links are perfect, zero-loss, infinite-bandwidth pipes.

- The Limitation: It cannot simulate dirty optics, CRC errors, optical power degradation, or latency spikes caused by fiber patch panel issues.
- The Impact: A routing architecture that relies on BiDirectional Forwarding Detection (BFD) with aggressive timers (e.g., 50ms) might work perfectly in Containerlab. In production, slight physical layer micro-flaps cause BFD to constantly tear down sessions, creating network instability.

3.3 Scale and Topology Constraints

Running a 1:1 digital twin of a massive datacenter (e.g., 10,000 nodes) in Containerlab is computationally impossible.

- The Limitation: You are forced to test a scaled-down subset of the topology (e.g., 2 spines, 4 leaves).
- The Impact: The simulation cannot predict how a BGP route-reflector architecture will handle the memory and CPU load of 100,000 routes in a full mesh, nor can it predict the fan-out impact of a broadcast storm in a massive L2 domain.

The greatest danger of the digital twin is the "It worked in the sim" syndrome. When engineering teams treat the twin as an oracle rather than a tool, they bypass human review and push changes directly to production. When the twin's abstraction omits a hardware specific or a dynamic failure state, the production network suffers a catastrophic outage that "passed all CI/CD checks."

A Mythos-compliant architecture does not abolish digital twins; it contextualizes them. The twin is one layer of a three-layer verification model.

5.1 Multi-Tool Triangulation

Do not rely solely on Batfish or Containerlab. Use them in tandem.

- Use Batfish for rapid syntactic validation and steady-state reachability assertions in CI/CD.
- Use Containerlab for complex protocol convergence testing and dynamic failure injection.

5.2 Physical Canaries (The Data-Plane Truth)

The ultimate digital twin is a physical testbed.

- Before pushing a radical ASIC-dependent feature (like MACsec or hardware PBR) to 1,000 switches, push it to a pair of physical switches in a lab rack.
- Generate real traffic. Measure the ASIC behavior. The physical canary is the only truth.

5.3 Formal Verification (Math > Simulation)

Instead of relying on running OSes, model the critical safety invariants (e.g., "The DMZ can never route to the DB") using formal verification languages like TLA+. A mathematical proof of correctness does not rely on parsing text or Linux dataplanes; it proves the logic is fundamentally sound.

Digital twins are invaluable for accelerating network automation and catching typos. But they are abstractions, and abstractions lie by omission. Batfish will lie about dynamic state. Containerlab will lie about ASIC behavior. True architectural safety requires understanding the exact limitations of the simulation layer, and always anchoring the final truth in the physical reality of the production canary.

```
A parser is not a router.
A container is not an ASIC.
A steady-state model is blind to failure.

If the twin is treated as an oracle, the network is a prototype.

> Simulation may be fast.
> Physical reality must be absolute.
```

End of Research Note RES-005

9. Source Register

Source	Authority and Status	Freshness Control
Batfish Documentation https://batfish.readthedocs.io/	Primary project documentation Living Published: Living	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.
Containerlab Documentation https://containerlab.dev/manual/	Primary project documentation Living Published: Living	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.
RFC 8342 - Network Management Datastore Architecture (NMDA) https://www.rfc-editor.org/rfc/rfc8342	Primary standard Stable RFC Published: 2018	Validated: 2026-07-22 Review on material revision, withdrawal, supersession, or scheduled report review.

10. Lineage, Review, and Publication Record

Record	Value
Original source file	RES-005_Network_Digital_Twin_Limitations_Batfish_Containerlabs_Constraints.md
Original source words	1325
Upgrade type	Enterprise research report; source and freshness validation; limitations; adoption decision; reproducibility plan
Generated on	2026-07-22
Current status	Candidate Research Report
Publication authority	Formal Mythos approval not yet recorded
Next review	180 days or event-driven trigger, whichever occurs first