SEAS-8414 CYBER ANALYTICS

Digital Twin & Remediation Simulation

Twin Construction, Scenario Replay & Blast Radius Analysis

Dr. Mallarapu · Breakwater Security Platform

What Phases 1-5 Built

The foundation for live network simulation

CHAPTER TAKEAWAY

Not simple. Here is why

ENRICHMENT VALUE

How do you sequence that?

PROGRESSIVE ANALYTICAL ARCHITECTURE

Phase 1
What is on the network?
573 tests

Phase 2
What is inside each device?
791 tests

Phase 3
How do devices behave?
2,101 tests

Phase 4
How do risks compound?
2,309 tests

Phase 5
Can we exploit it?
2,450 tests

The Phase 5 Limitation

Graph abstraction vs live simulation

CHAPTER TAKEAWAY

This is Chapter 6. Simulation analytics

ENRICHMENT VALUE

Chapter 1: descriptive. What exists?

✔ What Graph Analysis Provides

- **Static attack path computation**
  Yen's k-shortest paths on topology graph
- **Risk scoring (BRS)**
  Weighted multi-factor score per device
- **What-if remediation simulation**
  Graph edge removal and rescore
- **MITRE ATT&CK mapping**
  Tactic/technique labels on graph edges

VS

? What It Cannot Provide

- **Live traffic behavior under remediation**
  Will the patch break device communication?
- **Cascading failure prediction**
  Does isolating a switch cause downtime?
- **Pre-deployment validation**
  Test firewall rules before production push
- **Blast radius with real protocols**
  How far does compromise actually spread?

CHAPTER TAKEAWAY

Security teams are excellent at finding vulnerabilities

ENRICHMENT VALUE

Maintenance windows barely exist

> Before we rotated those credentials, we simulated it on a network twin with live protocol replay. Zero downtime. Full blast radius mapped.

- Remediation action planning
- Digital twin execution
- Topology-faithful replica
- Phase 3 transcript injection
- Cascade-free validation
- BFS propagation analysis

Phase 6 Architecture

Six-component pipeline: Build, Plan, Replay, Remediate, Cascade, Validate

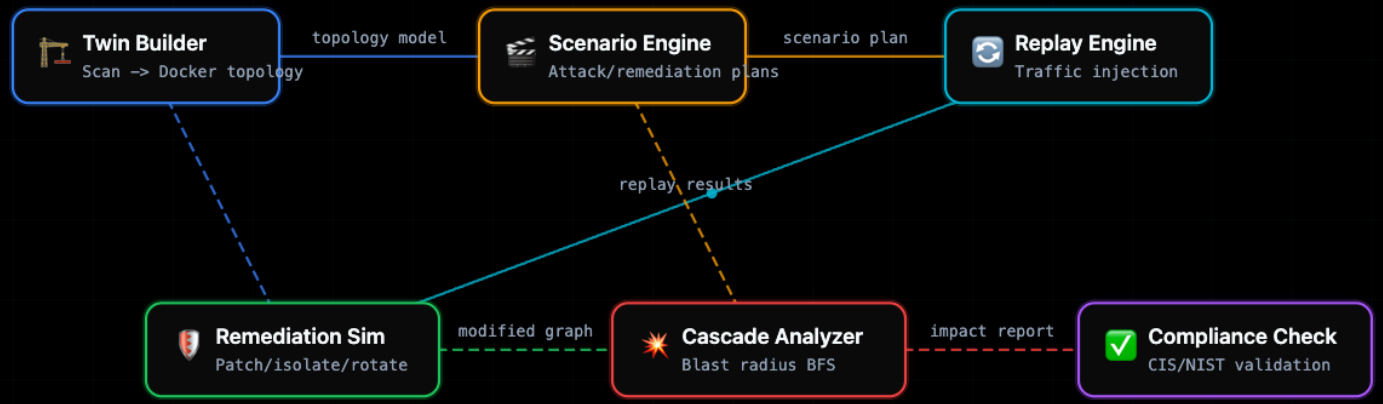
CHAPTER TAKEAWAY

Reason one: irreversible consequences

ENRICHMENT VALUE

Simulation provides the testing that these constraints demand

Phase 6 constructs a Docker-based digital twin from scan results, enabling safe attack replay and remediation testing without touching production. The twin builder generates containers from host profiles, the scenario engine executes attack and remediation plans, and the cascade analyzer verifies that fixes do not introduce downstream disruptions. Compliance checks validate that the post-remediation state meets NIST/IEC standards.



SECTION 01

Digital Twin Taxonomy

Origins, Categories, and Cybersecurity Applications

Digital Twin Origins

From manufacturing simulation to cybersecurity defense

CHAPTER TAKEAWAY

A software model that mirrors a production network

ENRICHMENT VALUE

Changes are tested in the twin before touching production

- **2003** — **Grieves Concept**
  Michael Grieves proposes "Mirrored Spaces Model" at U. of Michigan PLM course
- **2010** — **NASA Adoption**
  NASA uses digital twins for spacecraft lifecycle management and simulation
- **2012** — **GE Industrial**
  General Electric deploys twins for jet engine predictive maintenance
- **2017** — **Gartner Top 10**
  Digital twins named a top strategic technology trend
- **2020** — **CyberSec Twins**
  First network digital twins for attack simulation and defense validation
- **2024** — **Breakwater Twin**
  Scan-driven Docker topology with protocol transcript replay

Grieves Three-Component Model

Physical entity, virtual entity, and the data connection between them

CHAPTER TAKEAWAY

Twins exist on a fidelity spectrum

ENRICHMENT VALUE

Will disabling Telnet break the data path? That is topology

Physical Entity

The real-world system being modeled: network devices, topology, traffic patterns

Virtual Entity

The digital representation: Docker containers, simulated services, virtual links

Data Connection

Bidirectional link: scan data feeds twin, twin results inform remediation

NASA Digital Twin Definition

An integrated multi-physics, multi-scale, probabilistic simulation

CHAPTER TAKEAWAY

Five stages. Build, Deploy, Scenario, Remediate, Validate

ENRICHMENT VALUE

It is a living model, not a snapshot

> *"A digital twin is an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available models, sensor data, and fleet history to mirror the life of its corresponding physical twin."*
>
> -- NASA Technology Roadmap, 2010

Geometric Fidelity

Accurate topology representation

```
Scan-derived subnet/VLAN layout
```

Physical Fidelity

Behavior matches real system

```
Protocol transcript replay
```

Temporal Fidelity

Time-accurate response modeling

```
Latency-preserving traffic injection
```

Degradation Fidelity

Failure modes are realistic

```
Vulnerability exploitation + cascade
```

Cybersecurity Twin Categories

Three categories of digital twins for network security

CHAPTER TAKEAWAY

Let me paint the picture of the Clearwater plant

ENRICHMENT VALUE

That is our test case for today

Topology Twin

Models network structure: devices, links, subnets, VLANs. Static representation of connectivity.

- Asset inventory graphs
- Network maps
- Firewall rule sets

Behavioral Twin

Captures how devices communicate: protocol patterns, traffic flows, timing characteristics.

- Traffic baselines
- Protocol state machines
- Service dependencies

Simulation Twin

Executable replica that responds to inputs: attack replay, remediation testing, what-if analysis.

- Docker network twins
- Cyber ranges
- Attack simulators

Twin Category Comparison

Fidelity, cost, and capability tradeoffs across three categories

CHAPTER TAKEAWAY

CVE-2023-3595

ENRICHMENT VALUE

The question is deployment, not availability

DIMENSION	Topology Twin	Behavioral Twin	Simulation Twin
Data Source	Asset inventory, scan results	Traffic captures, flow logs	All above + device profiles
Fidelity	Structure only	Structure + timing	Full protocol replay
Executable	No (static graph)	Partial (replay only)	Yes (live containers)
Attack Testing	Path analysis	Anomaly detection	Full exploit replay
Update Freq	Per scan cycle	Continuous	On-demand build
Cost	Low (in-memory)	Medium (storage)	High (compute)

Breakwater Twin Positioning

Where we sit in the digital twin landscape

CHAPTER TAKEAWAY

How do you deploy a firmware patch to 40 controllers

ENRICHMENT VALUE

Not in theory. In simulation

🌍 Existing Approaches

Skybox Security
Topology twin only: asset model + firewall rule analysis. No live simulation.

Cyber Range (e.g. SimSpace)
Full simulation but manual setup: days of configuration per scenario.

AttackIQ / SafeBreach
Breach simulation on production: risk of disruption, no isolation.

SCADA Twins (e.g. Digital Bond)
ICS-specific, requires vendor-supplied device models.

VS

🎯 Breakwater Approach

Scan-driven auto-construction
Phases 1–5 scan data automatically generates Docker Compose twin

Protocol-faithful replay
Phase 3 transcripts injected into twin for behavioral validation

Isolated simulation environment
Docker bridge network, no production impact, full blast radius analysis

Integrated remediation testing
What-if engine patches applied in twin, validated before production push

Device Type to Twin Profile

23 device types mapped to 7 container profiles

CHAPTER TAKEAWAY

The twin consumes outputs from every prior chapter

ENRICHMENT VALUE

The twin is only as good as the data from earlier phases

Linux Server	server, nas, print-server, dns-server	`alpine + openssh + nginx`
Network Device	router, switch, access-point, firewall, gateway	`vyos/openwrt image`
IoT Camera	camera, nvr, doorbell, baby-monitor	`rtsp-server + http-admin`
IoT Sensor	thermostat, sensor, smart-plug, hvac	`mqtt-broker + http-api`
Workstation	desktop, laptop, thin-client	`ssh + smb + http`
Mobile/IoT	phone, tablet, smart-speaker, tv	`mdns + ssdp responder`
Industrial	plc, scada-hmi, rtu	`modbus-tcp + http-hmi`

Firewall Rule Inference

Scan open ports become twin ACL rules via iptables

CHAPTER TAKEAWAY

Here is what we cover today

ENRICHMENT VALUE

All through the lens of the water plant

acl_generator.sh — BASH

```bash
# Inferred ACL from scan open ports
# Device: twin-camera-42 (172.28.86.42)

# Allow discovered services
iptables -A INPUT -p tcp --dport 554 -j ACCEPT   # RTSP
iptables -A INPUT -p tcp --dport 80  -j ACCEPT   # HTTP
iptables -A INPUT -p tcp --dport 443 -j ACCEPT   # HTTPS

# Block everything else (closed in scan)
iptables -A INPUT -p tcp --dport 22  -j DROP      # SSH closed
iptables -A INPUT -p tcp --dport 23  -j DROP      # Telnet closed

# Default deny
iptables -A INPUT -j DROP
```

Twin Builder: Worked Example

20-device home network to executable Docker twin

CHAPTER TAKEAWAY

Four steps to build a twin

ENRICHMENT VALUE

Step 1: profile mapping. What is each device?

1. Ingest scan	IN: 20 hosts from Phase 1-5	OUT: HostData[20]
2. Profile mapping	IN: 20 hosts + device_type field	OUT: 4 cameras, 3 servers, 2 routers, 11 IoT
3. Subnet grouping	IN: 192.168.86.0/24 (20 devices)	OUT: 1 TwinSubnet -> 172.28.86.0/24
4. ACL generation	IN: 142 open ports across 20 devices	OUT: 142 ACCEPT + 20 default-DENY rules
5. Compose generation	IN: TwinTopology model	OUT: docker-compose.twin.yml (22 services)
6. Validation	IN: docker compose up --dry-run	OUT: All 22 containers pass health checks

2.3s	22	1.8 GB	162
BUILD TIME	CONTAINERS	EST. MEMORY	ACL RULES

SECTION 03

Scenario Engine

Attack Replay, Custom Scenarios, and Execution Pipelines

Three Scenario Types

Attack replay, remediation testing, and custom scenarios

CHAPTER TAKEAWAY

Tier 1: device type lookup

ENRICHMENT VALUE

"modbus" on a nonstandard port still maps to industrial

Attack Replay

Replay MITRE ATT&CK technique sequences against the twin to measure blast radius and time-to-compromise.

TRIGGER
MITRE technique IDs + entry point

OUTPUT
Compromised hosts, lateral paths, timing

Remediation Test

Apply a proposed fix (patch, ACL change, credential rotation) and verify no service disruption.

TRIGGER
What-if action from Phase 4

OUTPUT
Service availability, broken dependencies

Custom Scenario

User-defined YAML scenario combining attack steps and remediation actions in arbitrary order.

TRIGGER
YAML scenario definition file

OUTPUT
Full execution trace with diffs

MITRE ATT&CK Replay

Six-step attack sequence executed against the digital twin

CHAPTER TAKEAWAY

The 40 PLCs: device type "plc" maps to telnet-vuln

ENRICHMENT VALUE

62 devices, all mapped

Initial Access	Exploit Public App	T1190	twin-camera-42:80
Execution	Command Injection	T1059	twin-camera-42:shell
Discovery	Network Service Scan	T1046	172.28.86.0/24
Lateral Movement	Default Credentials	T1078	twin-nvr-10:22
Collection	Video Capture	T1125	twin-nvr-10:554
Exfiltration	Exfil Over C2	T1041	twin-router-1:443

Scenario Definition Format

YAML-based scenario specification with ATT&CK technique references

CHAPTER TAKEAWAY

confidence = 0.5 * type_match + 0.3 * port_overlap + 0.2 * service_hint

ENRICHMENT VALUE

Unknown device with no matching ports: 0.1 (floor)

camera-compromise-lateral.yml — YAML

```yaml
# scenario: camera-compromise-lateral.yml
name: "Camera Compromise with Lateral Movement"
description: "Exploit camera, pivot to NVR, exfiltrate video"
entry_point: "twin-camera-42"
timeout_seconds: 300

steps:
  - name: exploit_camera
    type: attack
    technique: T1190
    target: "twin-camera-42:80"
    payload: "cve-2023-28808"
    expected: "shell_access"

  - name: scan_subnet
    type: discovery
    technique: T1046
    target: "172.28.86.0/24"
    ports: [22, 80, 554, 8080]

  - name: lateral_to_nvr
    type: attack
    technique: T1078
    target: "twin-nvr-10:22"
    credentials: "admin:admin123"

  - name: exfiltrate_video
    type: collection
```

Production vs Twin Pentest

Why replay attacks on a twin instead of production

CHAPTER TAKEAWAY

Each /24 subnet becomes a SubnetBridge

ENRICHMENT VALUE

SCADA, corporate, DMZ

⚠ Production Pentest

Risk of service disruption
Exploit may crash production devices

Scheduling constraints
Must coordinate maintenance windows

Limited repeatability
Cannot re-run same exploit cleanly

Legal/compliance barriers
Some networks prohibit active testing

VS

✅ Twin Replay

Zero production impact
Isolated Docker network, disposable containers

On-demand execution
Build twin and replay in seconds, any time

Full repeatability
Rebuild twin, replay exact same scenario

Safe for regulated environments
No packets touch production network

Custom Scenario Builder

Five action block types compose into arbitrary scenario flows

CHAPTER TAKEAWAY

Category 1: per-port inbound rules

ENRICHMENT VALUE

Water plant: 120 per-port + 67 inter-host = 187 rules

ATTACK

Exploit or credential attack against a twin device

- CVE exploit
- Brute force
- Default creds

DISCOVERY

Network scanning or service enumeration step

- Port scan
- Service fingerprint
- ARP sweep

REMEDIATION

Apply a fix: patch, ACL change, credential rotation

- Apply patch
- Add ACL rule
- Rotate password

VALIDATION

Check service availability after remediation

- Health check
- Port verify
- Service probe

OBSERVATION

Capture metrics: traffic, latency, error rates

- Packet capture
- Latency measure
- Log collect

Scenario Execution Pipeline

Six-stage pipeline from YAML to execution report

CHAPTER TAKEAWAY

Misses Modbus master-slave between PLCs and HMIs

ENRICHMENT VALUE

Traffic replay and manual annotation help

The scenario engine processes attack and remediation plans through a six-stage pipeline. YAML scenario definitions are validated and resolved, Docker containers are spun up with health pre-checks, and each scenario step is executed with full timing instrumentation. After execution, logs, packet captures, and state diffs are collected into a structured JSON report for analysis.

EXECUTION FLOW

Scenario Comparison

Four remediation strategies tested against the same attack scenario

CHAPTER TAKEAWAY

Devices are nodes. Rules are edges

ENRICHMENT VALUE

Segmentation analysis: compare edge sets before and after

SCENARIO	COMPROMISED	DISRUPTED	BLAST RADIUS	TIME-TO-COMPROMISE
Baseline (no remediation)	8	0	12	0.8 min
Patch camera CVE only	5	0	7	2.4 min
Patch + rotate creds	2	1	3	7.5 min
Patch + creds + segment	0	0	0	blocked

Scenario Trace: Camera Compromise

Full execution log from twin build to scenario completion

CHAPTER TAKEAWAY

40 PLC nodes in SCADA subnet

ENRICHMENT VALUE

Historian has a cross-subnet edge

00:00.0	Twin build: 20 containers on 172.28.86.0/24	All healthy
00:02.3	Pre-check: 142 ports verified open	142/142 pass
00:03.1	Step 1: Exploit CVE-2023-28808 on twin-camera-42:80	Shell obtained
00:04.7	Step 2: Port scan 172.28.86.0/24 (54 ports)	18 hosts, 97 services
00:12.4	Step 3: SSH twin-nvr-10 with admin:admin123	Login success
00:15.8	Step 4: RTSP stream capture twin-nvr-10:554	60s captured
01:16.2	Scenario complete: 2/20 compromised, blast radius = 5	Criteria met

76.2s	2 / 20	5 devices	Yes
TOTAL TIME	COMPROMISED	BLAST RADIUS	CRITERIA MET

Blast Radius

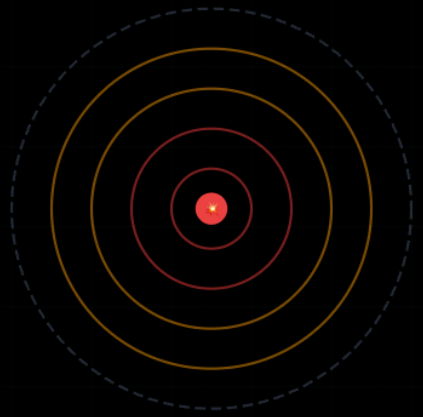
How far does a single compromise reach?

CHAPTER TAKEAWAY

sync_from_scan() for incremental updates

ENRICHMENT VALUE

Keeps the twin accurate over time



DEFINITION

The **blast radius** of a compromised device is the set of all devices that can be transitively reached through service dependencies, shared credentials, or network adjacency.

WHY IT MATTERS

A camera with CVSS 9.8 may seem like the top priority -- but if its blast radius is 1 device, while a printer with CVSS 6.5 can cascade to 15 devices, the printer is the bigger risk.

Service Dependency Graph

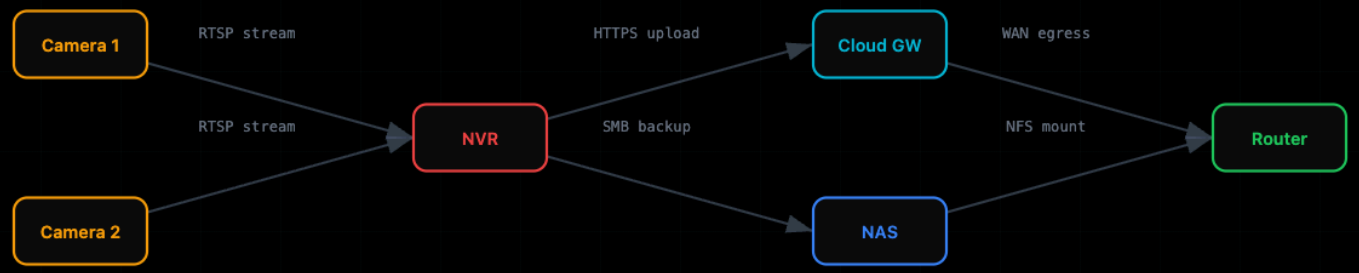
Directed edges show which devices depend on which services

CHAPTER TAKEAWAY

62 devices, 3 subnets, 187 rules

ENRICHMENT VALUE

Now we have something to simulate against



Propagation Probability Model

Probability decays with each hop through the dependency graph

CHAPTER TAKEAWAY

Purpose 1: validate twin fidelity

ENRICHMENT VALUE

Both are necessary before remediation

Hop-Decay Propagation

Base propagation formula: edge probability times depth-based decay

1

```
P(hop) = P_edge * decay^depth
```

Shared credentials: near-certain propagation

2

```
P_edge(shared_creds) = 0.95
```

where P_edge = 0.95

Same subnet without segmentation

3

```
P_edge(network_adj) = 0.70
```

where P_edge = 0.70

Each hop reduces probability by 15%

4

```
decay = 0.85 (configurable)
```

where decay = 0.85

After 3 hops via shared creds: 58.3% chance

5

```
P(depth=3) = 0.95 * 0.85^3 = 0.583
```

SECTION 04

Blast Radius Measurement

BFS Propagation, Probability Decay, and Impact Visualization

Measurement Methodology

Six-step process from compromise source to blast radius metrics

CHAPTER TAKEAWAY

SHA-256 hash of seed + target + technique + step count

ENRICHMENT VALUE

Same inputs, same results, every time

1. **Select compromise source**
   Choose the initially compromised device from twin topology
2. **Build dependency graph**
   Construct directed graph of service dependencies + credentials + adjacency
3. **Initialize BFS queue**
   Seed queue with source device, probability = 1.0, depth = 0
4. **Propagate with decay**
   For each neighbor: P_next = P_edge * decay^depth. Enqueue if P > cutoff
5. **Collect affected set**
   All visited devices with P > cutoff form the blast radius
6. **Compute metrics**
   Count devices, max depth, aggregate probability, services disrupted

Blast Radius Algorithm

BFS with probability-decay propagation and configurable cutoff

CHAPTER TAKEAWAY

Step 1: Telnet on PLC-07. Critical. Success

ENRICHMENT VALUE

3 devices compromised directly

blast_radius.py — PYTHON

```python
def compute_blast_radius(
    graph: nx.DiGraph,
    source: str,
    decay: float = 0.85,
    cutoff: float = 0.10,
) -> BlastRadiusResult:
    """BFS with probability decay."""
    queue = deque([(source, 1.0, 0)])   # (device, prob, depth)
    visited: dict[str, float] = {source: 1.0}
    max_depth = 0

    while queue:
        device, prob, depth = queue.popleft()
        max_depth = max(max_depth, depth)

        for neighbor in graph.successors(device):
            if neighbor in visited:
                continue

            edge = graph.edges[device, neighbor]
            p_edge = edge["propagation_probability"]
            p_next = p_edge * (decay ** depth)

            if p_next >= cutoff:
                visited[neighbor] = p_next
                queue.append((neighbor, p_next, depth + 1))
```

Blast Radius Visualization

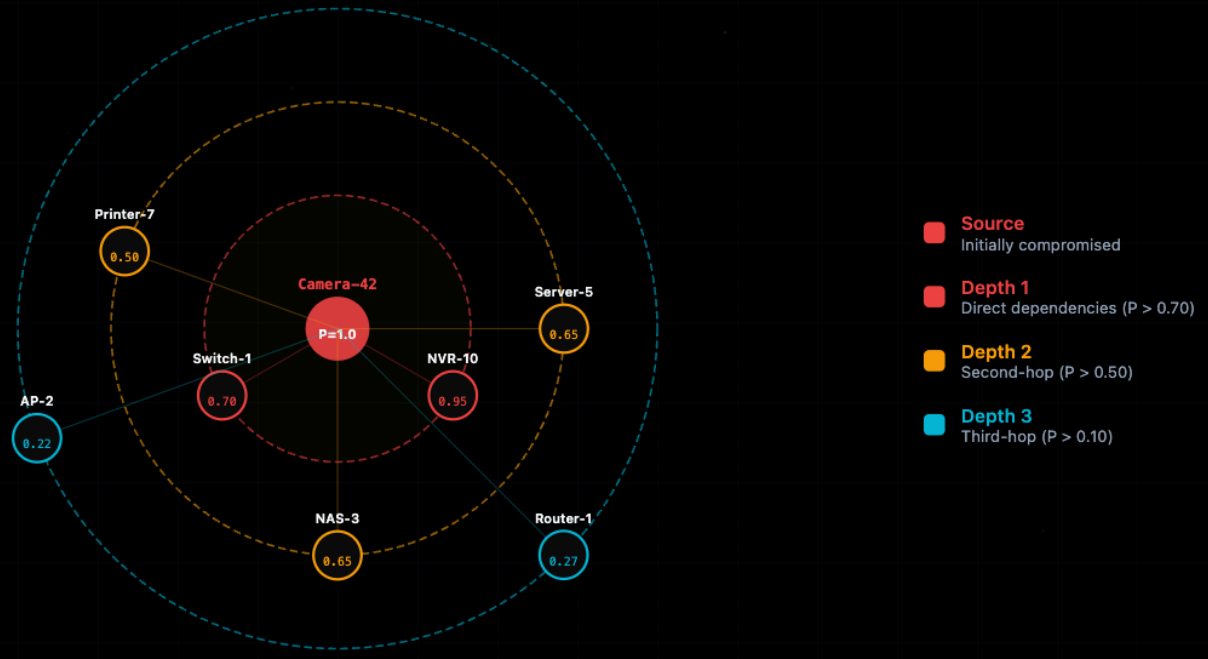
Concentric rings show propagation depth from Camera-42 compromise

CHAPTER TAKEAWAY

3 compromised. Now check firewall rules

ENRICHMENT VALUE

Almost a third of the network



Source
Initially compromised

Depth 1
Direct dependencies (P > 0.70)

Depth 2
Second-hop (P > 0.50)

Depth 3
Third-hop (P > 0.10)

Segmentation Effect on Blast Radius

Before and after VLAN segmentation in the digital twin

CHAPTER TAKEAWAY

Checkpoint, apply, cascade check, risk score, validate, preview

ENRICHMENT VALUE

All six happen automatically when you call apply_remediation

🔴 Flat Network (Before)

Single /24 subnet
All 20 devices on 192.168.86.0/24 -- full L2 adjacency

Camera-42 blast radius: 12
Compromise propagates to 12 of 20 devices

Max depth: 4 hops
Attacker reaches router via camera -> NVR -> server -> router

Mean P: 0.54
High average compromise probability across network

SEG

🟢 Segmented (After)

Three VLANs: IoT, IT, Mgmt
Inter-VLAN routing with ACL: only NVR->Server:445 allowed

Camera-42 blast radius: 2
Compromise limited to IoT VLAN (camera + NVR only)

Max depth: 1 hop
ACL blocks cross-VLAN propagation at Layer 3

Mean P: 0.48
Only high-probability edges within the IoT segment

SECTION 05

Traffic Replay

Protocol Transcript Injection into the Digital Twin

Traffic Replay Architecture

Inject Phase 3 transcripts into twin, compare responses to production

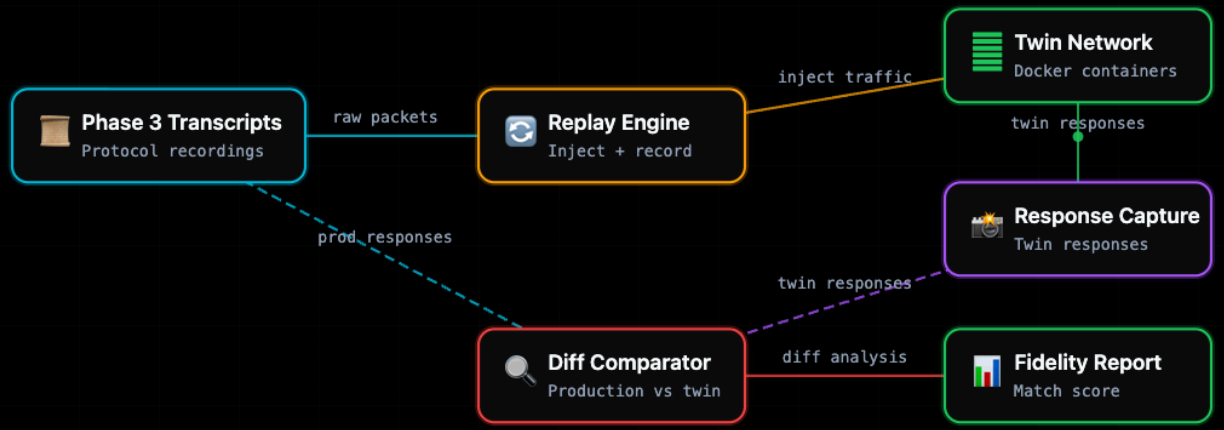
CHAPTER TAKEAWAY

patch_cve, rotate_credentials, segment_network

ENRICHMENT VALUE

Each modifies the topology in a specific way

Traffic replay validates twin behavioral fidelity by injecting real protocol transcripts from Phase 3 into the Docker twin and comparing its responses against production recordings. The diff comparator produces a fidelity score that quantifies how faithfully the twin mirrors real device behavior. A high match score (above 95%) confirms the twin is suitable for reliable attack simulation and remediation testing.



Transcript Data for Replay

Six fields extracted from Phase 3 recordings drive twin replay

CHAPTER TAKEAWAY

Port exposure: weighted by protocol risk

ENRICHMENT VALUE

Floor at zero per device

`protocol`	Phase 3 grammar inference	`HTTP/1.1, RTSP/1.0`	Select replay protocol handler
`request_bytes`	Protocol transcript recording	`GET /api/status HTTP/1.1\r\n...`	Raw bytes injected into twin
`response_bytes`	Protocol transcript recording	`HTTP/1.1 200 OK\r\n...`	Expected response for comparison
`timing_ms`	Transcript timestamp deltas	`12.4, 3.1, 45.7`	Replay with realistic timing
`source_ip`	Phase 1 discovery data	`192.168.86.42`	Map to twin container IP
`state_context`	Phase 3 state machine	`authenticated, stream_active`	Resume from correct protocol state

Twin Connection Client

IP-mapped replay with timing fidelity and response capture

CHAPTER TAKEAWAY

Before: ports 32 + firewall 4 + creds 20 + vulns 9 = 65

ENRICHMENT VALUE

Reduction: 47.5 points. 73%

twin_replay_client.py — PYTHON

```python
class TwinReplayClient:
    """Connect to twin devices using mapped IPs."""

    def __init__(self, topology: TwinTopology):
        # Build IP mapping: production -> twin
        self.ip_map: dict[str, str] = {}
        for subnet in topology.subnets:
            for device in subnet.devices:
                self.ip_map[device.original_ip] = device.twin_ip

    async def replay_transcript(
        self,
        transcript: ProtocolTranscript,
    ) -> ReplayResult:
        # Map production IP to twin container IP
        twin_ip = self.ip_map[transcript.source_ip]
        port = transcript.destination_port

        # Connect to twin device
        reader, writer = await asyncio.open_connection(
            twin_ip, port
        )

        results: list[ExchangeResult] = []
        for exchange in transcript.exchanges:
            # Inject request bytes
            writer.write(exchange.request_bytes)
            await writer.drain()
```

Production vs Twin Response Diff

Fidelity analysis comparing twin responses to Phase 3 transcripts

CHAPTER TAKEAWAY

Patch CVEs on 40 PLCs (40 actions)

ENRICHMENT VALUE

Total: 82 actions

DEVICE	PROTOCOL	EXCHANGES	MATCH	MISMATCHES	FIDELITY
Camera-42	HTTP	24	95.8%	Timestamp header, Content-Length (gzip)	HIGH
Camera-42	RTSP	8	100%	None	HIGH
NVR-10	HTTP	16	87.5%	Session cookie, CSRF token	MEDIUM
NVR-10	SSH	4	100%	None	HIGH
Router-1	HTTP	12	75.0%	Dynamic JS bundle hash, uptime counter	MEDIUM
Sensor-8	MQTT	32	100%	None	HIGH
Server-5	SMB	6	66.7%	Negotiate dialect, security blob	LOW

102	91.2%	4/7	1/7
TOTAL EXCHANGES	OVERALL MATCH	HIGH FIDELITY	NEEDS TUNING

Twin Fidelity Score

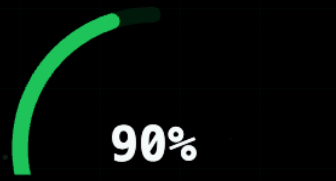
Quantifying how faithfully the digital twin reproduces real device behavior

CHAPTER TAKEAWAY

BRS before: 1,890. After: 645. Delta: -1,245

ENRICHMENT VALUE

Two cascading failures detected



Overall Fidelity

`F = SUM(w_i * s_i) = 0.900`

FIDELITY COMPONENTS

Anomaly Injection

Injecting malicious packets validates twin behavioral accuracy

CHAPTER TAKEAWAY

Historian loses Modbus path (segmentation)

ENRICHMENT VALUE

Both addressable

INJECTION TYPE	TWIN RESPONSE	REAL RESPONSE	MATCH
⚠ **Malformed Headers** Invalid protocol headers to test error handling paths	RST + error log	RST + error log	✔
⟳ **Replay Attack** Duplicate authenticated session tokens from prior capture	Session rejected	Session rejected	✔
☠ **Buffer Overflow** Oversized payload exceeding expected field length	Truncate + 400	Crash + restart	✘
✘ **State Violation** Send commands out of expected protocol state order	State error	State error	✔

Fidelity Drift Report

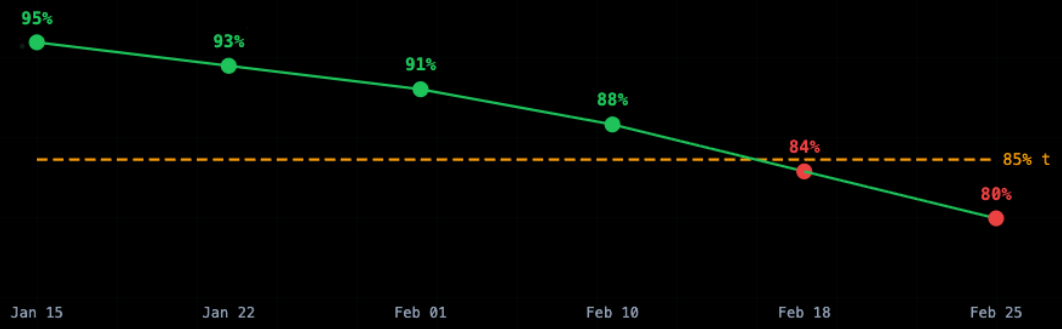
Twin accuracy degrades as real devices change -- drift signals re-calibration need

CHAPTER TAKEAWAY

Historian: add firewall exception for the data collection path

ENRICHMENT VALUE

Three iterations. Milliseconds each



DRIFT ROOT CAUSES

Firmware update on real device	-3.2%
New service endpoint added	-2.1%
Configuration policy change	-1.8%

Action: Re-calibrate twin from latest scan data

Why Simulate Before Deploying?

Real-world remediation failures are costly and dangerous

CHAPTER TAKEAWAY

Constrained scheduling

ENRICHMENT VALUE

Objective: minimize total time

⚠ Without Simulation

- **Credential rotation breaks NVR recording**
  HVAC cameras go dark for 4 hours
- **Firmware patch causes boot loop**
  Physical truck roll to recover device
- **Firewall rule blocks SCADA traffic**
  Production line halted, $50K/hr loss
- **Service disable breaks dependencies**
  Cascading failure across 12 devices

VS

✔ With Twin Simulation

- **Twin validates credential propagation**
  Pre-verify all dependent connections
- **Twin tests firmware compatibility**
  Catch boot loop in sandbox first
- **Twin models traffic flow impact**
  Verify SCADA traffic unaffected
- **Twin maps service dependencies**
  Zero-disruption deployment confidence

Checkpoint Mechanism

Snapshot complete device state before any remediation action

CHAPTER TAKEAWAY

Phase 1: extract dependencies

ENRICHMENT VALUE

Phase 3: execute groups sequentially

Network Config

- IP addresses
- VLAN assignments
- Firewall rules
- Routing tables

Credentials

- Username/password pairs
- API tokens
- Certificate chains
- SSH keys

Service State

- Running services
- Port bindings
- Protocol versions
- Firmware version

Health Baseline

- Response times
- Error rates
- Uptime counters
- Connection counts

Checkpoint Rollback

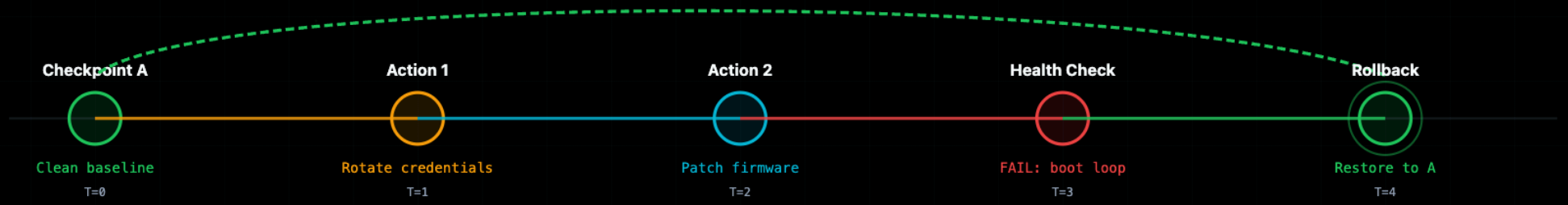
When remediation fails health checks, automatically restore to last known good state

CHAPTER TAKEAWAY

14 groups of 3 PLCs

ENRICHMENT VALUE

65% time reduction



Rollback restores full checkpoint state in < 100ms on the twin

Five Remediation Action Types

Each action type is simulated differently on the digital twin

CHAPTER TAKEAWAY

Simulator replays the ordering step by step

ENRICHMENT VALUE

Operations team adjusts and retries

Credential Rotation	Replace default or shared credentials with unique, strong ones	`Update auth state, re-test all dependent sessions`	RISK Low
Firmware Patch	Apply vendor security update to eliminate known vulnerabilities	`Swap protocol behavior model, validate service continuity`	RISK Medium
Service Disable	Turn off unnecessary services (telnet, FTP, debug ports)	`Remove service from twin, check dependency graph`	RISK Medium
Network Segmentation	Isolate device into a dedicated VLAN or firewall zone	`Update reachability matrix, re-compute attack paths`	RISK High
Firewall Rule	Block specific inbound traffic at network or host firewall	`Filter incoming connections, verify legitimate traffic unaffected`	RISK Low

Network Segmentation Simulation

Model VLAN isolation on the twin -- verify attack paths are severed

CHAPTER TAKEAWAY

Reboot time: log-normal, mean 135s

ENRICHMENT VALUE

Dependency completeness: Bernoulli(0.85)

BEFORE: FLAT NETWORK

AFTER: SEGMENTED VLANS

VLAN 10: OT	Camera	PLC
VLAN 20: IT	Workstation	ERP Server
VLAN 30: Security	NVR	
VLAN 40: Guest	Guest WiFi	

Attack paths reduced: 9 to 2 (cross-VLAN paths eliminated)

Risk Score Delta

Before/after BRS comparison shows remediation effectiveness

CHAPTER TAKEAWAY

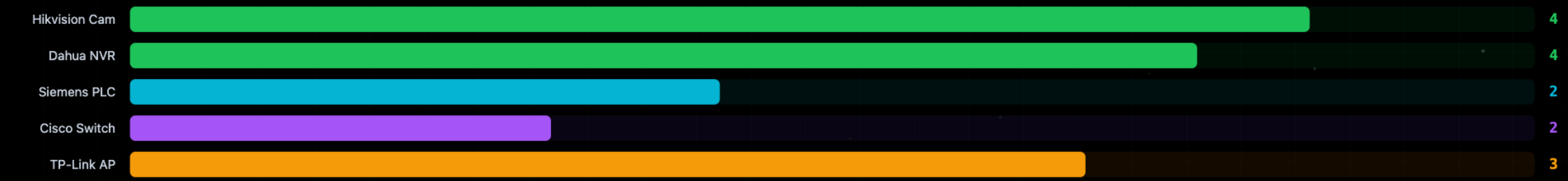
BRS delta: mean -1,245, 95% CI [-1,380, -1,110]

ENRICHMENT VALUE

Total time: mean 2,340s, 95% CI [1,890, 3,120]

⚠	✔	↓	⬇
6	3	−3	4
Avg BRS Before	Avg BRS After	Avg Delta ↓	Max Reduction

BRS REDUCTION PER DEVICE



Risk Delta Visualization

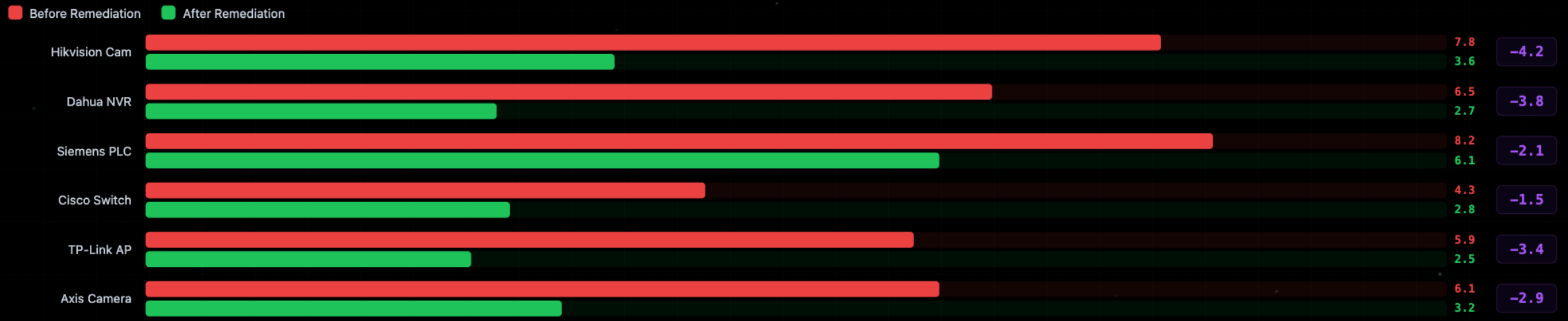
Side-by-side BRS comparison shows remediation impact per device

CHAPTER TAKEAWAY

One in fifty executions might breach the constraint

ENRICHMENT VALUE

Worth it? Absolutely



Cascade Severity Metrics

Quantifying the cascade risk for each remediation action type

CHAPTER TAKEAWAY

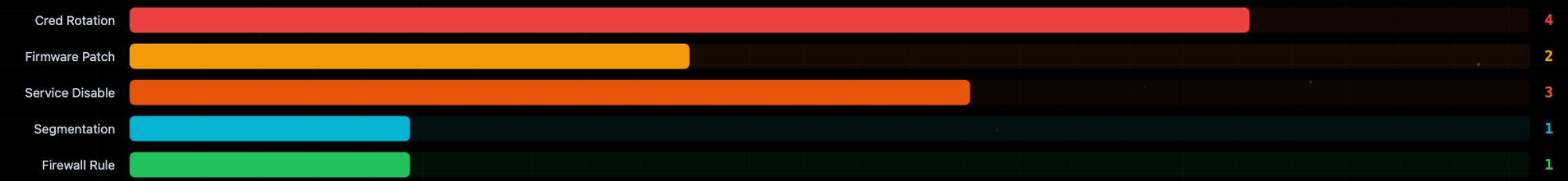
After patching: PLC response times shift

ENRICHMENT VALUE

Traffic volume doubles without explanation? Investigate

23	6	2	4
Actions Tested	Cascades Detected	Blocked Actions	Max Affected

AVG CASCADE SCORE BY ACTION TYPE



Prediction Calibration Loop

Twin results feed back into graph weights for increasingly accurate predictions

CHAPTER TAKEAWAY

Build twin: 62 devices, 3 subnets, 187 rules

ENRICHMENT VALUE

Deploy: 47 minutes, no disruption

1. **Graph predicts BRS delta**
   Fast computation using edge weights and path analysis
2. **Twin simulates action**
   Empirical execution with health checks and cascade detection
3. **Compare predicted vs actual**
   Compute prediction error and direction (over/under)
4. **Update edge weights**
   Adjust graph weights to minimize prediction error
5. **Re-predict with calibrated graph**
   Next prediction is more accurate from learned weights

↻ Loop

Graph-Twin Agreement Score

Quantifying how well graph predictions align with twin simulation results

CHAPTER TAKEAWAY

Finding vulnerabilities is necessary but not sufficient

ENRICHMENT VALUE

Zero-disruption validation verifies continuity

Agreement Score Derivation

1. Base agreement formula (per action)

```
A = 1 − (|predicted − actual| / max_delta)
```

2. Graph-computed BRS change

```
predicted = graph_brs_after − graph_brs_before
```

3. Twin-measured BRS change

```
actual = twin_brs_after − twin_brs_before
```

4. Global agreement across all simulated actions

```
A_global = mean(A_i) for all actions i
```

5. If agreement drops below 85%, re-calibrate graph weights

```
Calibration trigger: A_global < 0.85
```


Current Agreement 0.91